



**OPTIMAL COLLISION AVOIDANCE TRAJECTORIES FOR
UNMANNED/REMOTELY PILOTED AIRCRAFT**

DISSERTATION

Nathan E. Smith, Colonel, USAF

AFIT-ENY-DS-14-D-34

**DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY**

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the United States Air Force, the Department of Defense, or the United States Government.

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

OPTIMAL COLLISION AVOIDANCE TRAJECTORIES FOR
UNMANNED/REMOTELY PILOTED AIRCRAFT

DISSERTATION

Presented to the Faculty
Graduate School of Engineering and Management
Air Force Institute of Technology
Air University
Air Education and Training Command
in Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy

Nathan E. Smith, B.S.E.E., M.S.A.E
Colonel, USAF

December 2014

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

OPTIMAL COLLISION AVOIDANCE TRAJECTORIES FOR
UNMANNED/REMOTELY PILOTED AIRCRAFT

Nathan E. Smith, B.S.E.E., M.S.A.E
Colonel, USAF

Approved:

<u>/signed/</u>	<u>27 Oct 2014</u>
Richard G. Cobb, PhD (Chairman)	Date
<u>/signed/</u>	<u>23 Oct 2014</u>
William P. Baker, PhD (Member)	Date
<u>/signed/</u>	<u>23 Oct 2014</u>
David R. Jacques, PhD (Member)	Date
<u>/signed/</u>	<u>23 Oct 2014</u>
John F. Raquet, PhD (Member)	Date

Accepted:

<u>/signed/</u>	<u>27 Oct 2014</u>
A. B. Badiru	Date
Dean, Graduate School of Engineering and Management	

Abstract

The post-911 environment has punctuated the force-multiplying capabilities that Remotely Piloted Aircraft (RPA) provides combatant commanders at all echelons on the battlefield. Not only have unmanned aircraft systems made near-revolutionary impacts on the battlefield, their utility and proliferation in law enforcement, homeland security, humanitarian operations, and commercial applications have likewise increased at a rapid rate. As such, under the Federal Aviation Administration (FAA) Modernization and Reform Act of 2012, the United States Congress tasked the FAA to “provide for the safe integration of civil unmanned aircraft systems into the national airspace system (NAS) as soon as practicable, but not later than September 30, 2015.” However, a necessary entrance criterion to operate RPAs in the NAS is the ability to Sense and Avoid (SAA) both cooperative and non-cooperative air traffic to attain a target level of safety as a traditional manned aircraft platform. The goal of this research effort is twofold: First, develop techniques for calculating optimal avoidance trajectories, and second, develop techniques for estimating an intruder aircraft’s trajectory in a stochastic environment. This dissertation describes the optimal control problem associated with SAA and uses a direct orthogonal collocation method to solve this problem and then analyzes these results for different collision avoidance scenarios.

*To my beautiful “Proverbs 31” wife, thank you for your love that is everlasting.
To our sons, thank you for the joy you bring.
To my parents, thank you for everything.*

Acknowledgments

I want to express my heartfelt appreciation to my research advisor, Dr. Rich Cobb, for his incredible encouragement, guidance, and leadership throughout the course of this Ph.D. program. I am truly thankful for his friendship and the countless hours he spent helping me to understand, develop, write, and present the research.

Likewise, I want to thank my research committee members, Dr. Baker, Dr. Jacques, and Dr. Raquet, for their outstanding support and guidance.

I am also indebted to Maj Scott Pierce for his help in providing a greater understanding of stochastic estimation techniques. He also graciously co-authored two conference papers with me, which both appear in this document.

I am grateful to Maj Chris Arendt for his help in developing the conditional inequality constraints in this document. I am also thankful for his help as a co-author on a manuscript, which appears in this document.

Finally, I want to express my sincere thanks to Dr. Robert Chen for his help in the model development and analysis of simulations results.

Nathan E. Smith

Table of Contents

	Page
Abstract	iv
Dedication	v
Acknowledgments	vi
Table of Contents	vii
List of Figures	xii
List of Tables	xvii
List of Symbols	xviii
List of Acronyms	xx
 I. Introduction	 1
1.1 Background	1
1.1.1 Limitations with Current Methods for Access into the NAS	3
1.1.2 USAF RPA Operations	4
1.2 Current SAA Research	5
1.2.1 System Description	6
1.2.2 Areas for Improvement	7
1.3 Research Motivation for Optimal Control & Stochastic Estimation	7
1.3.1 Research Goal	8
1.3.2 Research Questions	8
1.4 Expected Contributions	9
1.5 Assumptions & Limitations	9
1.6 Dissertation Overview	10
 II. Background	 12
2.1 Sensors	12
2.1.1 Sensor Types	13
2.1.1.1 Cooperative Sensors	13
2.1.1.2 Non-Cooperative Sensors	13
2.1.1.3 MIAA Sensor Suite Overview	14
2.1.2 Definitions and Terminology	15
2.1.2.1 Pairwise vs. Concurrent Avoidance	15
2.1.2.2 Global vs. Local Avoidance	16

	Page
2.1.2.3 Collision vs. Obstacle Avoidance	17
2.2 Literature Review	19
2.2.1 Collision Avoidance	19
2.2.2 Stochastic Estimation	21
2.2.3 Dynamic Models	23
2.2.3.1 Intruder Model	23
2.2.3.2 Ownship Model	25
2.2.3.3 Look-Ahead Time Horizon	26
2.3 Research Approach	27
III. Optimal Control Methodology	29
3.1 Methodology	29
3.1.1 Direct Orthogonal Collocation	29
3.1.2 Receding Horizon	30
3.1.3 Stochastic Estimation Method	31
3.2 Optimal Control Problem Formulation	31
3.3 Deterministic 2D Optimal Control Problem	34
3.3.1 Minimize Time	34
3.3.2 Minimize Path Deviation	36
3.4 2D Results	37
3.4.1 Case 1: Single Intruder, “Nose-to-nose” Geometry, Minimize Time .	38
3.4.2 Case 2: Single Intruder, “Crossing” Geometry, Minimize Time . . .	39
3.4.3 Case 3: Single Intruder, “S-Turn” Geometry, Minimize Time	41
3.4.4 Case 4: Single Intruder, “Nose-to-nose” Geometry, Minimize Path Deviation	42
3.4.5 Case 5: Two Intruder, Accelerating “Nose-to-nose” and “Crossing” Geometry, Minimize Path Deviation	44
3.5 Summary of Optimal Control Results	45
IV. Estimation Methodology	47
4.1 3D Model Description	47
4.2 Particle Filter Development	48
4.3 3D Scenario Description and Implementation	50
4.3.1 Scenario Description	51
4.3.2 State Propagation	51
4.3.3 Observation Model	52
4.4 3D Results	54
4.5 Conclusion	61
V. Uncertainty Corridors for 3D Collision Avoidance	62
5.1 Model Development	62
5.1.1 Intruder Model Development	63

	Page
5.1.1.1 Coordinated Turn Model for Horizontal Plane	63
5.1.1.2 Singer Acceleration Model for Vertical Motion	66
5.1.2 Observation Model	67
5.2 2D Coordinated S-Turn Scenario	68
5.3 Collision Avoidance Constraint Modeling	71
5.3.1 Algorithm Considerations and Development	71
5.3.2 Algorithm Overview	72
5.3.3 Generating the Distribution	73
5.3.4 Capture Probability via Minimum Volume Enclosing Ellipsoid . . .	74
5.3.5 Ensuring Correct Ellipsoid Orientation	76
5.4 Interpolation Algorithm	78
5.5 Optimal Control Problem	81
5.5.1 Problem Formulation	81
5.5.2 Stochastic 3D Scenario Description	82
5.6 3D Stochastic Simulation Results	83
5.7 Analysis of Results	89
5.8 Conclusion	93
VI. Implementing Conditional Inequality Constraints for Optimal Collision Avoidance	94
6.1 Introduction	94
6.2 Conditional Inequality Constraint Approximation Methods	97
6.3 Minimum Area Enclosing Superellipse (MAES)	98
6.4 Sigmoid Function	102
6.4.1 Sigmoid Sum Method	103
6.4.2 Sigmoid Product Method	105
6.5 Description of Example Problems	109
6.6 Constraints	110
6.7 Performance Measure	111
6.8 Example Problem 1	111
6.9 MAES Simulation Results	112
6.10 Sigmoid Simulation Results	115
6.10.1 Sigmoid Sum Results	115
6.10.2 Sigmoid Product Results	115
6.11 Sensor Tolerance Evaluation Results	115
6.12 Example Problem 2	119
6.13 Right of Way Formulation	120
6.13.1 Simulation Results and Analysis	122
6.14 Conclusion	124
VII. Comparison of Trajectory Optimization Methods	126
7.1 Model Parameters	127
7.2 Algorithm Description	129
7.2.1 JOCA Description	129

	Page
7.2.2 Optimal Control Description	130
7.3 Simulation Methodology	131
7.4 Scenario Description	132
7.5 Results	135
7.5.1 Scenario One	136
7.5.2 Scenario Two	139
7.5.3 Scenario Three	142
7.5.4 Scenario Four	145
7.6 Summary and Analysis of Results	148
VIII.Nonlinear Filter Development and Comparison	153
8.1 Observation and Dynamics Models for Filter Evaluation	154
8.2 Filter Description	157
8.2.1 Van Loan Method	158
8.2.2 EKF Development	161
8.2.3 UKF Development	162
8.2.4 PF Development	163
8.3 Evaluation Methodology	167
8.4 Observability Grammian	168
8.5 Observability Results	169
8.6 Scenario Description	173
8.7 Nonlinear Filter Results	173
8.7.1 Scenario One, Case One: Standard Rate Turn	174
8.7.2 Scenario One, Case Two: Half Standard-Rate Turn	178
8.7.3 Scenario One, Case Three: No Turn	180
8.7.4 Scenario Two, Case One: Standard-Rate Turn	181
8.7.5 Scenario Two, Case Three: No-Turn	182
8.7.6 Scenario Three, Case One: Standard-Rate Turn	183
IX.Overlap Criteria between Cylinder and Ellipsoid	188
9.1 Background on Cylinder-Ellipsoid Constraint Development	188
9.2 Cylinder-Ellipsoid Constraint Development	190
9.3 Cylinder-Ellipsoid Example Problem	194
9.4 Scenario Description	195
9.5 Cylinder-Ellipsoid Scenario Results	198
9.6 Cylinder-Ellipsoid Scenario Observations and Conclusions	203
X. Conclusions and Recommendations	207
10.1 Optimal Control Future Research Recommendations	209
10.2 Optimal Estimation Future Research Recommendations	211

	Page
Appendix A:	213
Appendix B:	221
Appendix C:	226
Bibliography	229

List of Figures

Figure	Page
1.1 DOD Unmanned Aircraft System Flight Hours Per Year [1]	2
1.2 DOD Unmanned Aircraft System Group Designation [1]	4
1.3 AFRL’s Airborne Sense-and-Avoid System Overview [2]	6
1.4 Comparison of Different Sense and Avoid Solution Methods	8
2.1 Pairwise Versus Concurrent Collision Avoidance Solutions. Adapted from [3] .	16
2.2 Replan by Calculating New Optimal Solution for the Goal. Adapted from [4] .	18
2.3 Replan by Calculating Optimal Solution Back to Path. Adapted from [4] . . .	18
2.4 Intruder State Propagation Methods. Adapted from [3].	22
2.5 Forces Acting on the Aircraft. Adapted from [2].	26
3.2 Case 1, Separation	38
3.1 Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Intruder (Red) by 1,500 Feet Separation Distance While Minimizing Total Time.	39
3.3 Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Intruder (Red) by 1,500 Feet Separation Distance While Minimizing Total Time.	40
3.4 Case 2, Separation	40
3.5 Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Intruder Aircraft (Red) by 1,500 Feet Separation Distance While Minimizing Total Time.	41
3.6 Case 3, Separation Distance	41
3.7 Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Intruder Aircraft (Red) by 1,500 Feet Separation Distance While Minimizing Path Deviation.	43
3.8 Case 4, Separation	43
3.9 Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Both Intruder Aircraft #1 (Red) and Intruder Aircraft #2 (Green) by 1,500 Feet Separation Distance While Minimizing Path Deviation.	44

Figure	Page
3.10 Case 5, Separation	45
3.11 Uncertainty Grows Over Time	46
4.1 Time Series of Optimal Trajectory for Ownship (Blue) Avoiding the Intruder Aircraft's (Red) Uncertainty Volume While Minimizing Path Deviation with No Measurement Updates.	56
4.2 Time Series of Optimal Trajectory for Ownship (Blue) Avoiding the Intruder Aircraft's (Red) Uncertainty Volume While Minimizing Path Deviation, Showing the Effect of Measurement Updates at Time 18 and 30 Seconds for the Receding Horizon.	58
5.1 2D S-Turn Trajectory	70
5.2 SLIMVEE Flowchart	73
5.3 Point Cloud at 8 & 11 Seconds	74
5.4 Point Cloud Capture at 8 & 11 Seconds	76
5.5 Determining Correct Ellipsoid Orientation for Interpolation	77
5.6 MVEE's for 30-Second Left Turning Intruder Trajectory	78
5.7 Slerp Interpolation at 9.5 Seconds	80
5.8 Point Line Distance from [5].	82
5.9 Time Series for Ownship (Blue) Avoiding the Intruder's (Red) Probability Region with No Measurement Updates.	85
5.10 No Measurement Updates	86
5.11 Time Series for Ownship (Blue) Avoiding the Intruder's (Red) Probability Region with Measurement Updates.	87
5.12 Separation Distance with Measurement Updates	89
5.13 Optimal Path Comparison	90
5.14 Closest Point of Approach	92
6.1 Approximating Conditional Inequality Path Constraints	100
6.2 Horizontal and Vertical Separation Sigmoid Functions	102

Figure	Page
6.3 3D Constraint Contour Comparison of Sigmoid Sum and Product Methods . .	108
6.4 Time Series of Optimal Trajectory for Ownship (Blue) Avoiding the Intruder Aircraft (Red) While Minimizing Path Deviation (MAES, Sigmoid Results Similar).	113
6.5 Results Using $N = 200$ for MAES Approximation of Inequality Path Constraint	114
6.6 Time Series of Optimal Trajectory for Ownship (Blue) Avoiding the Intruder Aircraft (Red) by Adhering to Right of Way While Minimizing Path Deviation.	122
7.1 Scenario One	133
7.2 Scenario Two	134
7.3 Scenario Three	134
7.4 Scenario Four	135
7.5 Results from Scenario One	136
7.6 Scenario One - Time Series of Baseline Optimal Trajectory for Ownship (Blue) Avoiding Intruder (Red) While Minimizing Path Deviation.	137
7.7 Distance from Intruder 1	138
7.8 Results from Scenario Two	139
7.9 Scenario Two - Time Series of Baseline Optimal Trajectory for Ownship (Blue) Avoiding Both Intruders (Red) While Minimizing Path Deviation.	141
7.10 Separation Distance from Scenario Two	142
7.11 Results from Scenario Three	143
7.13 Distance from Intruder 1	143
7.12 Scenario Three - Time Series of Baseline Optimal Trajectory for Ownship (Blue) Avoiding Intruder (Red) While Minimizing Path Deviation.	144
7.14 Results from Scenario Four	146
7.15 Separation Distance from Scenario Four	146
7.16 Scenario Four - Time Series of Baseline Optimal Trajectory for Ownship (Blue) Avoiding Intruders (Red) While Minimizing Path Deviation.	147

Figure	Page
8.1 Particle Filter Algorithm. Adapted from [6, 7]	166
8.2 2D Scenario for Observability Analysis	170
8.3 Reduced Areas of Observability Based on Geometry	171
8.4 Observability Grammian Condition Number and Singular Values at 1 Hz . . .	171
8.5 Observability Analysis Mean & Worst-Case Position Errors	172
8.6 Observability Analysis Mean & Worst-Case Turn-Rate Error	172
8.7 Mean & Worst-Case position Error for Standard-Rate Turn	175
8.8 Close-In of Mean Position Errors	175
8.9 Position Errors Above 820-Foot Threshold	176
8.10 Mean & Worst-Case Turn-Rate Error for Standard-Rate Turn	177
8.11 Mean CPU Time	178
8.12 Mean & Worst-Case Position Error for Half Standard-Rate Turn	179
8.13 Mean & Worst-Case Turn-Rate Error for Half Standard-Rate Turn	180
8.14 Mean & Worst-Case Position Error for No Turn	181
8.15 Mean & Worst-Case Turn-Rate Error for No Turn	181
8.16 Abeam Geometry: Mean & Worst-Case Position Error for Standard-Rate Turn	182
8.17 Abeam: Mean & Worst-Case Turn-Rate Error for Standard-Rate Turn	183
8.18 Abeam Geometry: Mean & Worst-Case Position Error for No Turn	183
8.19 Abeam: Mean & Worst-Case Turn-Rate Error for No Turn	184
8.20 Mean & Worst-Case Position Error for Standard-Rate Turn	184
8.21 Mean & Worst-Case Turn-Rate Error for Standard-Rate Turn	185
8.22 Mean CPU Time	185
9.1 Superellipsoid for Varying Values of Exponential ϵ_1, ϵ_2 . Adapted from [8] . . .	192
9.2 Superellipsoid Approximation of Cylinder for Varying Values of n	193
9.3 Initial Geometry for Cylinder-Ellipsoid Scenario	197
9.4 Initial Geometry and Uncertainty Volumes for Cylinder-Ellipsoid Scenario . . .	199
9.5 Cylinder-Ellipsoid Scenario at Time 15 Seconds	200

Figure	Page
9.6 Cylinder-Ellipsoid Scenario at Time 25.7 Seconds	202
9.7 Cylinder-Ellipsoid Scenario at Time 45 Seconds	204
B.1 Mean & Worst-Case Position Error for Half Standard-Rate Turn	221
B.2 Mean & Worst-Case Turn-Rate Error for Half Standard-Rate Turn	222
B.3 Mean & Worst-Case Position Error for Half Standard-Rate Turn	223
B.4 Mean & Worst-Case Turn-Rate Error for Half Standard-Rate Turn	223
B.5 Mean & Worst-Case Position Error for No Turn	224
B.6 Mean & Worst-Case Turn-Rate Error for No Turn	225

List of Tables

Table	Page
2.1 SAA Sensor Characteristics [9]	15
3.1 Summary of Simulation Scenarios	38
4.1 Initial Particle Value Distribution	52
4.2 Minimum Intruder 3D Separation Distance	60
6.1 Results for MAES Approximation of Conditional Inequality Constraint	113
6.2 Results for Sigmoid Sum Approximation of Conditional Inequality Constraint .	115
6.3 Results for Sigmoid Product Approximation of Conditional Inequality Constraint	115
6.4 Comparison of Methods for Achieving Error Less Than Sensor Tolerance	117
7.1 Summary of Percentage Difference from Baseline Results	148
7.2 Summary of Optimal Control NLP Execution Times for Scenario Two	151
9.1 Intruder Initial Conditions for Cylinder-Ellipsoid Scenario	196
9.2 NLP Execution Times for Cylinder-Ellipsoid Scenario	203
A.1 Survey Results	214

List of Symbols

Symbol	Definition
\mathbf{A}	Ellipsoid characterization matrix
\mathbf{C}	Time-varying path
\mathbf{F}	Jacobian matrix for the dynamics function $\mathbf{f}(\mathbf{x}(t))$
\mathbf{F}_{x_k, z_k}	Discrete-time state transition matrix
\mathbf{Q}	Weighted state deviation matrix
\mathbf{Q}_k	Discrete-time noise strength
\mathbf{R}	Rotation matrix
\mathbf{R}_m	Measurement noise autocorrelation
\mathbf{R}_o	Weighted control usage matrix
\mathbf{S}	Boundary inequality constraint
\vec{e}	Unit eigenaxis vector
\mathbf{g}	Inequality constraint
\mathbf{q}	Quaternions
\mathbf{u}	Control
\mathbf{x}	State trajectory
J	Performance measure
L	Cost function or number of states
N	Number of particles or superellipse exponential
N_z	Normal acceleration
S	Sigmoid value
T_{CPA}	Time to closest point of approach (CPA) (sec)
V	Ground speed (ft/sec)
a_z	Acceleration in z-axis (ft/sec ²)
g	Gravity constant (ft/sec ²)

Symbol	Definition
h	Horizontal constraint (ft)
s	Speed or sigmoid stiffness
v	Vertical constraint (ft)
v_x	Velocity in x-axis (ft/sec)
w	Particle weight
w	Additive white Gaussian noise
Δxy	Horizontal separation distance (ft)
Δz	Vertical separation distance (ft)
χ_k	Particle set
σ	Diagonal matrix of eigenvalues
χ	Heading angle (radians)
δ	Overestimation error
ϵ	Sigmoid exponential
γ	Flight path angle (radians)
λ	Lagrange multiplier or optimization parameter
μ	Bank angle (radians)
ω	Turn-rate (rad/sec)
Φ	State transition matrix
ϕ	Terminal cost function
ψ	Equality constraint
τ	Time transformation
θ	Rotation angle or relative azimuth angle (radians)

Subscripts

k	Discrete-time index
0	Initial value

List of Acronyms

Acronym	Definition
ABSAA	Airborne Sense and Avoid
ADS-B	Automatic Dependent Surveillance-Broadcast
AESA	Active Electronically Scanned Array
AFRL	Air Force Research Laboratory
ATA	Air Transport Association
ATC	Air Traffic Control
CA	constant acceleration
COA	Certificate of Waiver or Authorization
CONUS	Continental United States
CPA	closest point of approach
CV	constant velocity
DHS	Department of Homeland Security
DOD	Department of Defense
EKF	Extended Kalman Filter
EO	electro-optical
FAA	Federal Aviation Administration
gPC	Generalized Polynomial Chaos
GPS	Global Positioning System
HALE	High Altitude Long Endurance
HLS	Homeland Security
JOCA	Jointly Optimal Collision Avoidance
MAES	minimum area enclosing superellipse
MDP	Markov Decision Process
MIAA	Multiple Intruder Autonomous Avoidance
MPC	Model Predictive Control

Acronym	Definition
MSL	Mean Sea Level
MVEE	minimum volume enclosing ellipsoid
NAS	National Airspace System
OCP	Optimal Control Problem
OSD	Office of the Secretary of Defense
PDF	probability density function
PDDD	Program Design Description Document
POMDP	Partially Observable Markov Decision Process
RA	resolution advisory
RHC	Receding Horizon Control
ROW	right of way
RPA	Remotely Piloted Aircraft
RRT	Rapidly-exploring Random Tree
RTCA	Radio Technical Commission for Aeronautics
SAA	Sense and Avoid
SADL	Situational Awareness Data Link
SDE	Stochastic Differential Equation
SIR	Sequential Importance Resampling
SQP	Sequential Quadratic Programming
TCAS	Traffic Collision Avoidance System
TLS	Target Level of Safety
UAS	Unmanned Aircraft Systems
UKF	Unscented Kalman Filter
USAF	United States Air Force

OPTIMAL COLLISION AVOIDANCE TRAJECTORIES FOR UNMANNED/REMOTELY PILOTED AIRCRAFT

I. Introduction

THE POST-911 environment has punctuated the force-multiplying capabilities that Unmanned Aircraft Systems (UAS) provide combatant commanders at all echelons on the battlefield. With over a dozen years of relentless combat operations, “commanders have come to rely upon robust and persistent support based on unmanned platforms to execute their core missions against hostile forces” [1].

Not only have UAS made near-revolutionary impacts on the battlefield, their utility and proliferation in law enforcement, Homeland Security (HLS), humanitarian operations, and commercial applications have likewise increased at a rapid rate [10]. For example, the Department of Homeland Security (DHS) used Predator (MQ-1) Remotely Piloted Aircraft (RPA) to conduct surveillance operations of the international border between United States and Mexico. Further, Global Hawk (RQ-4) aircraft have aided in humanitarian relief efforts following a devastating earthquake in Haiti and a catastrophic tsunami in Japan. A recent report to Congress expressed the potential applications for unmanned aircraft systems are “bound only by human ingenuity” [11].

1.1 Background

Since 2002, the number of Department of Defense (DOD) UAS went from 200 to over 11,000 [10]. Likewise, as seen in Figure 1.1, the cumulative number of DOD flight hours by unmanned systems went from approximately 50,000 hours in 2002 to over one million flight hours in May 2010, and six-months later, surpassed one million combat hours in November 2010 [1]; and this number continues to climb.

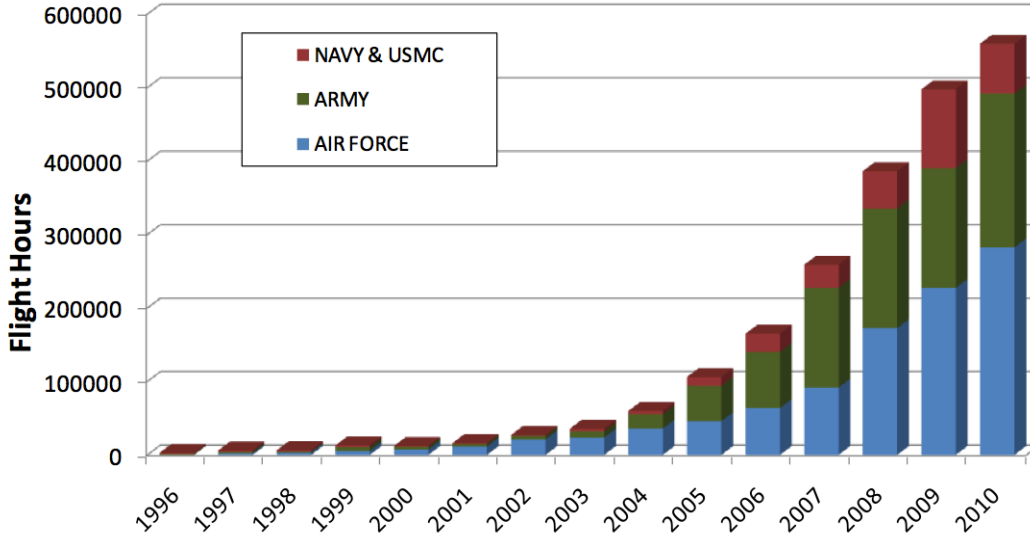


Figure 1.1: DOD Unmanned Aircraft System Flight Hours Per Year [1]

The United States Air Force (USAF) clearly recognizes the utility and explosive growth of RPAs and has in recent years increased its RPA pilot training despite the drawdown in combat operations [12].¹ Senior policy makers clearly understand that the near insatiable desire and need to operate RPAs in the Continental United States (CONUS) and overseas will continue into the foreseeable future. As such, under the Federal Aviation Administration (FAA) Modernization and Reform Act of 2012, the United States Congress tasked the FAA to “provide for the safe integration of civil unmanned aircraft systems into the National Airspace System (NAS) as soon as practicable, but not later than September 30, 2015” [13].

Without a traditional pilot in the cockpit, “One of the biggest challenges for integration of UAS into the NAS is the loss of the ability to see and avoid other aircraft, as required by Title 14 Code of Federal Regulations” [14]. Although the FAA Modernization and Reform Act of 2012 does not specify particular technological solutions, this Act plainly acknowledges

¹Note: The term UAS refers to the entire unmanned aircraft system to include the ground control stations, the communication infrastructure, and the aircraft. The term RPA refers strictly to the physical aircraft and not to the associated infrastructure required to support flight operations.

that a necessary entrance criterion to safely operate RPAs in the NAS and overcome the “see and avoid” limitation is the necessity for RPAs to Sense and Avoid (SAA) both cooperative and non-cooperative air traffic. The goal of this research is to support the DOD and FAA efforts to safely integrate and normalize RPA operations in the NAS.

1.1.1 Limitations with Current Methods for Access into the NAS.

Current methods for access into the NAS for RPA operations are “greatly limited under interim FAA policies” [1]. In order to conduct RPA operations “outside of restricted, warning, and prohibited areas” the DOD and other “federal, state, or local agencies” must receive authorization from the FAA “under a (temporary) Certificate of Waiver or Authorization (COA)” [11, 15]. The COA contains the operating capabilities, limitations and technical maturity of the unmanned system and is a formal request to the FAA for approval to conduct specific flight operations [11]. The FAA reviews each COA and then imposes various operating restrictions or limitations that the FAA deems necessary “to ensure the [RPA] can operate safely with other airspace users” [16]. Despite the recent improvement to speed up the arduous and time-consuming COA process, this process still “does not provide the level of airspace access necessary to accomplish the wide range of DOD [RPA] missions at current and projected operational tempos (OPTEMPOs)” [15]. The Office of the Secretary of Defense (OSD) Unmanned Systems Roadmap [15] goes on to say that the airspace “constraint” in the United States “will only be exacerbated as combat operations” overseas drawdown and unmanned systems return to stateside locations where RPA operators will inevitably commence their required training and proficiency operations.

Following the COA process, the current methods segregate, not “integrate”, unmanned from manned aircraft operations in the NAS. These methods either involve procedural restrictions such as temporary no-fly zones which separate manned aircraft from unmanned aircraft or require chase pilots in manned aircraft to fly in close proximity to an unmanned aircraft in order to “see and avoid” for the RPA. Both of these methods unnecessarily constrict and tax an already limited airspace resource. Further, the additional complexity, increased safety risk, and added cost of manned chase aircraft do not necessarily add tangible

safety benefits. For instance, RPA pilots operate and fly the Global Hawk by entering aircraft maneuver commands via a computer keyboard and mouse and not via a traditional joystick; therefore, the Global Hawk cannot immediately respond to chase pilot commands to maneuverer to avoid an oncoming threat. Further, pilots in civil aircraft are flying chase missions at night for Reaper and Predator RPAs in order to comply with FAA COA guidance. RPA experts familiar with chase procedures agree these night chase missions increase the risk to both the chase pilots and other airspace users.

1.1.2 USAF RPA Operations.

The USAF has invested a significant portion of its UAS resources into Group 4 and 5 unmanned aircraft systems (reference Figure 1.2 for UAS Group designation). For the Air Force, these systems consist of the MQ-1A/B Predator, the MQ-9 Reaper, and the High Altitude Long Endurance (HALE) RQ-4 Global Hawk.



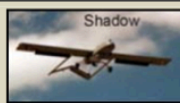


UAS Groups	Maximum Weight (lbs) (MGTOw)	Normal Operating Altitude (ft)	Speed (kts)	Representative UAS	
Group 1	0 – 20	<1200 AGL	100	Raven (RQ-11), WASP	
Group 2	21 – 55	<3500 AGL	< 250	ScanEagle	
Group 3	< 1320	< FL 180		Shadow (RQ-7B), Tier II / STUAS	
Group 4	>1320	< FL 180	Any Airspeed	Fire Scout (MQ-8B, RQ-8B), Predator (MQ-1A/B), Sky Warrior ERMP (MQ-1C)	
Group 5		> FL 180		Reaper (MQ-9A), Global Hawk (RQ-4), BAMS (RQ-4N)	

Figure 1.2: DOD Unmanned Aircraft System Group Designation [1]

With the exception of climbs or descents, most Group 5 UAS operations in the NAS occur in Class A airspace, which extends above 18,000 feet Mean Sea Level (MSL) to 60,000 feet MSL [1]. By regulation, Air Traffic Control (ATC) actively monitors and provides positive separation between all aircraft in Class A airspace [1]. Because Class A airspace is under “positive control”, the region of increased concern for collision avoidance systems are operations outside Class A airspace (such as climbs and descents) where ATC is not responsible for monitoring and providing positive separation between all aircraft in the airspace.

1.2 Current SAA Research

In recent years, triggered by the explosive growth in RPAs, the scientific and research communities have invested considerable efforts into researching SAA technologies to allow RPAs to “integrate” into the NAS. Among such efforts are those by the Air Force Research Laboratory (AFRL), Massachusetts Institute of Technology’s Lincoln Laboratory, and others as described in the next chapter. The body of work presented herein focuses on and supports the ongoing SAA development efforts of AFRL.

Since 2008, AFRL and its industry partners have been working on a multi-sensor Airborne Sense and Avoid (ABSAA) system for the Global Hawk “under the Multiple Intruder Autonomous Avoidance (MIAA) science and technology program” [17]. The goal of this program is to prevent mid-air collisions by enabling an “unmanned aircraft to autonomously detect and avoid both cooperative and noncooperative intruders, responding within minutes to longer-range threats to maintain safe separation, and within seconds to short-range threats to avoid collisions” [17]. This system uses electro-optical sensors and radar to detect and track noncooperative targets such as general-aviation aircraft without transponders and uses the Automatic Dependent Surveillance-Broadcast (ADS-B) and Traffic Collision Avoidance System (TCAS) to sense and track cooperative air traffic [17].

Since 2010, AFRL’s ABSAA system has conducted a series of flight tests using a Learjet equipped with the MIAA sensors and algorithms and acting as a surrogate for the unmanned Global Hawk [17]. These test flights evaluated the collision-avoidance algorithms and a new

electronically scanned sense-and-avoid radar, as well as a new technique to perform passive target ranging from the two-dimensional imagery provided by electro-optical sensors [17]. Preliminary results from these tests have helped to refine the operation of the collision avoidance algorithms and improve the operation and integration of the airborne sensors used to “sense” air traffic.

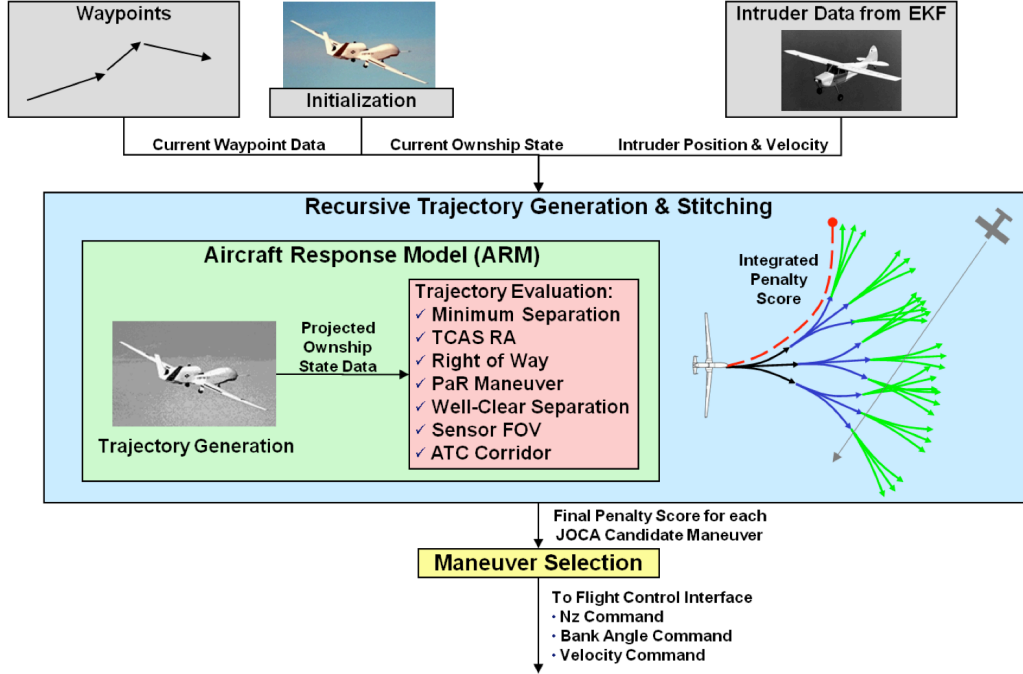


Figure 1.3: AFRL’s Airborne Sense-and-Avoid System Overview [2]

1.2.1 System Description.

AFRL’s ABSAA system initializes with the RPA’s current position and planned route of flight. The system then projects the RPA’s position 30-seconds into the future and calculates a series of distinct collision avoidance trajectories based on pre-determined fixed increments of control modeled as normal acceleration (N_z) and bank angle (μ) changes. Within the aircraft response model, the algorithm has a hierarchy of penalties or costs used to score each of the collision avoidance trajectories (depicted by the colored branches extending from the RPA in Figure 1.3). The algorithm then selects the trajectory with

the lowest score, and if required, commands the RPA to fly this lowest cost trajectory. Figure 1.3 from [2] shows a high-level overview of AFRL’s ABSAA system. AFRL and its industry partner’s Program Design Description Document (PDDD) [2] contains additional system details.

1.2.2 Areas for Improvement.

Preliminary analysis by AFRL and its industry partners indicates there are several areas for potential performance improvements in AFRL’s current ABSAA system design. These areas include quantifying the performance improvements by using an optimal control solution, implementing a higher performing nonlinear estimation algorithm to minimize intruder uncertainty, and developing more robust algorithms to predict an intruder’s trajectory in a stochastic environment. The body of work herein addresses each of these improvement areas.

1.3 Research Motivation for Optimal Control & Stochastic Estimation

Most real-time SAA algorithms (such as those developed by AFRL, Lincoln Laboratory, and others) rely on a dynamic programming method where the system designers force the algorithm to select the *best* collision avoidance trajectory from a limited set of predetermined or *canned* trajectories. In contrast, an optimal control method calculates only one avoidance trajectory which is the *optimal trajectory* based on a system designer’s specified optimality criteria. Figure 1.4 graphically depicts the difference between the dynamic programming method and the optimal control method.

The motivations for this research for posing the collision avoidance problem for RPAs as an optimal control problem are: (1) this formulation is flexible and can easily incorporate time constraints, path deviations, and control use into an appropriate cost functional to meet mission objectives while maintaining the required separation distance from one or more intruder aircraft; (2) these optimal solutions can provide design engineers a baseline or basis against which to judge other collision avoidance algorithms; and (3), design engineers can potentially use these optimal solutions in real-time RPA collision avoidance applications.

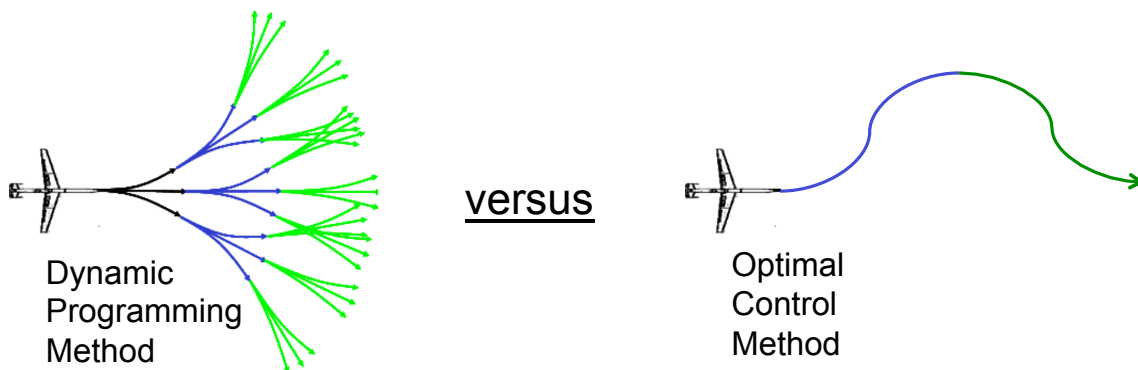


Figure 1.4: Comparison of Different Sense and Avoid Solution Methods

Additionally, since knowledge of the intruder's current and future position is unknown, solving the SAA optimal control problem requires application of stochastic estimation techniques. These techniques are necessary to estimate the intruder's position using imperfect sensor measurements and limited dynamics models. Thus, the motivation for applying stochastic estimation is to reduce the intruders' state error covariance and in turn increase the performance of the SAA algorithm.

1.3.1 Research Goal.

The goal of the body of work herein is to expand on AFRL's efforts and develop (1) techniques for calculating optimal collision avoidance trajectories for RPAs, and (2) techniques for estimating an intruder aircraft's trajectory in a stochastic environment. The body of work herein will demonstrate these results by *flying* these algorithms in a computer-based simulation.

1.3.2 Research Questions.

This research will answer the following three questions:

1. How do you formulate the airborne collision avoidance problem as an optimal control problem?

2. How should you model and estimate stochastic intruder(s) for an airborne collision avoidance application in the optimal control problem?
3. How do you account for ownship and intruder(s) uncertainties for an airborne collision avoidance application in the optimal control problem?

1.4 Expected Contributions

The expected contributions of this research is to add to AFRL’s toolset in assisting the DOD and FAA safely integrate and normalize RPAs operations in the NAS. Although this research focuses on an RPA with *Global Hawk-like* performance characteristics, the results and methodology are platform agnostic and transferable to any RPA platform.

1.5 Assumptions & Limitations

The FAA has expressed concern that integration of unmanned systems in the NAS would require RPAs achieving a Target Level of Safety (TLS) equivalent to manned aircraft operations [14]; however, to date, the FAA has not provided system designers a quantifiable metric for accessing an *acceptable* TLS for RPA NAS integration. The Radio Technical Commission for Aeronautics (RTCA) is a Federal Advisory Committee that is providing TLS recommendations to the FAA [18]. In particular, RTCA Special Committee-228, Minimum Operational Performance Standards for Unmanned Aircraft System, is working in a collaborative effort with the DOD and other federal and civil aviation agencies to define acceptable TLS criteria for RPA operations [19]. The preliminary results from this collaborative effort indicate that the TLS criteria will likely base minimum separation distance as a function of the closure rate between the RPA and the approaching air traffic. The work herein assumes the FAA will delineate a required TLS and identify a quantifiable metric for accessing TLS. In the interim, the work herein formulated TLS separation criteria based on MIAA program documentation [2], which used a separation distance that was not a function of closure rate. However, the problem formulation methodology in the work herein can accommodate any feasible TLS separation criteria to include a combination of time and distance. Finally, the focus of the work herein is to support typical NAS

operations. As such, the intruder dynamics model assumes nonaggressive maneuvers and that the intruder(s) are not adversarial, that is the intruder(s) do not intentionally try to collide with the unmanned aircraft.

1.6 Dissertation Overview

This dissertation contains ten chapters. This first chapter introduced the motivation and goals for this research. Chapter II provides a literature review of the current algorithms and stochastic models available to support airborne collision avoidance. Chapters III (optimal control) and IV (estimation) introduce the research methodology and provide a framework for answering the research questions. Chapter V presents a new method and algorithmic procedure for formulating uncertainty corridors in 3D for inclusion as time-varying inequality path constraints for the nonlinear optimal control problem. Chapter VI proposes two efficient methods to enforce conditional inequality path constraints in the optimal control problem formulation and compares and contrasts these approaches on representative airborne avoidance scenarios. Chapter VII is a limited comparison between the Jointly Optimal Collision Avoidance (JOCA) algorithm used in the Multiple Intruder Autonomous Avoidance (MIAA) program [2] and the optimal control algorithm developed through the work herein. The motivation for this comparison is to address the question “How do the trajectories of an optimal control approach compare to the JOCA algorithm results when generating an airborne collision avoidance solution?” Chapter VIII analyzes the simulation results for three different nonlinear estimation filters and compares their performance for use in an airborne sense and avoid application within the NAS. The motivation for this comparison is to address the question “How does the performance of these nonlinear filters compare when estimating and predicting an intruder’s current and future trajectories for the optimal airborne collision avoidance problem?” Chapter IX combines the developments in the earlier chapters to formulate a new inequality path constraint that keeps a cylindrical keep-out region around the ownship optimally away from an ellipsoidal probability region around an intruder and then demonstrates this formulation in a stochastic

multi-intruder scenario. Finally, Chapter X summarizes and lists recommendations for future research based on the body of work herein.

II. Background

THERE ARE many interrelated aspects that make the airborne Sense and Avoid (SAA) optimal control problem challenging and complex. This chapter cites different reviews in the literature that survey key SAA topics. Since a fundamental premise of the airborne SAA problem is the ability to detect or “sense” an intruder or threat aircraft, this chapter first introduces the different types of sensors used to perform this function. Next, prior to beginning the literature review, this chapter defines key terms and clarifies terminology for the work herein. This chapter then reviews various aspects of the airborne SAA problem and compares and contrasts current methods in the literature for the following three areas: (1) collision avoidance algorithms, (2) stochastic estimation techniques for collision avoidance applications, and (3) dynamic models to characterize aircraft response. Finally, this chapter concludes with a general overview of the research approach for the work herein.

2.1 Sensors

In order to avoid a collision, a SAA system must first have the ability to detect (that is, sense) an intruder or threat aircraft. The focus of this research effort is not on this sense or detect phase; however, an aspect of this research does focus on methods to incorporate stochastic sensor measurements into the collision avoidance algorithm, and thus, some discussion on sensors is warranted. Based on the sensor characteristics, this research models sensor performance to estimate measurement uncertainties and examines how these uncertainties impact the overall collision avoidance system’s performance. Therefore, the intent of this section is to merely introduce and provide a general overview of the different types of sensors available for the detect phase. For additional details, Maroney et al. [20] provides an overview of sensor requirements for UAS SAA development.

2.1.1 Sensor Types.

Initially sensors can be categorized as either cooperative or non-cooperative sensors. Both sensor types are capable of performing the sense or detect function; however, as noted by Chen et al. [9] there is currently not a single sensor that adequately addresses SAA requirements for both cooperative and non-cooperative traffic. As a result, a robust SAA system requires a variety of sensor types to include those able to detect cooperative and non-cooperative air traffic.

2.1.1.1 Cooperative Sensors.

There are two types of cooperative sensors: those that broadcast their position when queried by an integrator and those that continuously broadcast their position without requiring an interrogation. TCAS is an example of a query-based cooperative sensor and ADS-B as well as tactical data links such as Situational Awareness Data Link (SADL) and Link-16 are examples of continuously broadcasting cooperative sensors. The obvious advantage of cooperative sensors is these sensors provide information regardless of the ownship's position or geometry (that is, these sensors are not limited by field of view or by errors in the ownship's heading angle). Also, with cooperative sensors, the ownship and intruder can potentially communicate and mutually collaborate on an acceptable collision avoidance paths and thus eliminate a key uncertainty area in collision avoidance algorithms of determining intruder's intent. TCAS II algorithm employs a collaborative collision avoidance solution [21].

2.1.1.2 Non-Cooperative Sensors.

There are two types of non-cooperative sensors: active and passive. Non-cooperative active sensors transmit electronic emissions from the host platform whereas non-cooperative passive sensors do not transmit electronic emissions. Radar is an example of a non-cooperative active sensor and electro-optical (EO) is an example of a non-cooperative passive sensor. Non-cooperative sensors are necessary to detect aircraft that do not broadcast their position state information. Further, depending on the operational environment, a non-

cooperative passive sensor may be more advantageous than an active sensor in order to minimize enemy detection from radar emissions.

2.1.1.3 MIAA Sensor Suite Overview.

The sensor suite for the MIAA program, which this research supports, includes a mix of cooperative and non-cooperative sensors and consists of the following four sensors [2]: TCAS, ADS-B, EO, and radar. This diverse sensor array provides an excellent balance that maximizes detection across a wide-range of intruder types. The following paragraphs provide a short description of each sensor type.

2.1.1.3.1 TCAS. TCAS is a collision avoidance system that is required by FAA regulation to be installed on all large transport aircraft, both military and commercial. TCAS works by interrogating the transponders (Mode A, C, or S) of airborne aircraft in order to obtain range, bearing, and relative altitude information [9]. The range measurement for TCAS is calculated from the elapsed time for a response signal to return from the interrogated aircraft; bearing or azimuth is calculated via a directional antenna which measures the angle of return signal from the interrogated aircraft. TCAS provides different levels of traffic advisories to the pilots with the most severe being a resolution advisory (RA) which directs the pilot to perform a vertical avoidance maneuver in order to avoid hitting an intruder. TCAS does not command horizontal avoidance maneuvers. Furthermore, the system provides no collision avoidance protection against an aircraft that does not have an operating transponder [21].

2.1.1.3.2 ADS-B. ADS-B is the satellite-based successor to the FAA's aging ground-based surveillance radar system [22]. ADS-B uses Global Positioning System (GPS) to determine and then share precise aircraft location information in real-time with other ADS-B equipped aircraft, both manned and unmanned. By 1 January 2020, all aircraft flying in designated classes of airspace within the NAS must have a functional ADS-B system onboard.

2.1.1.3.3 EO. A strength of an EO sensor is that it typically provides highly accurate angular measurements at a reasonably fast update rate [9]. However, known

limitations of a single monoscopic EO sensor are (1) the sensor does not inherently generate range information and (2) typical detection ranges are poor. The MIAA program is developing a custom EO sensor suite specifically designed to perform the SAA detect function onboard an autonomous Group 5 UAS [23].

2.1.1.3.4 Radar. For over 60 years radar has provided detection for airborne targets. In particular the MIAA program utilizes an Active Electronically Scanned Array (AESA) radar specifically designed to accomplish the SAA detect function onboard an autonomous Group 5 UAS [24]. Because the observation model for the radar system is the most challenging of all the sensor models, the radar system is the primary sensor modeled for the collision avoidance scenarios in the work herein.

Table 2.1 from [9] list typical accuracies, update rates, and detection ranges for the four sensors described earlier.

Table 2.1: SAA Sensor Characteristics [9]

	TCAS	ADS-B	Radar	EO
Accuracy	Range: 175 - 300 ft Bearing: 9-15 deg Altitude: 50-100 ft	Horizontal position: 25 – 250 ft Vertical position: 50 – 100 ft	Azimuth: 0.5 - 2 deg Elevation: 0.5 - 2 deg Range: 10-200 ft Range rate: 1 - 10 ft/s	Azimuth: 0.1 - 0.5 deg Elevation: 0.1- 0.5 deg
Update rate	1 Hz	1 Hz	0.2 to 5 Hz	20 Hz
Detection range	≥ 14 nm	≥ 20 nm	5 - 10 nm	2 - 5 nm

2.1.2 Definitions and Terminology.

Prior to discussing the literature review, this section first defines key terms and clarifies terminology to establish a common framework for the body of work herein.

2.1.2.1 Pairwise vs. Concurrent Avoidance.

An important distinction to make in collision avoidance algorithms is how the formulation handles a multiple intruder environment. In this regard, there are two options the formulation can take: the first is pairwise, where the formulation computes an avoidance solution by sequentially resolving in a pairwise fashion each ownship-intruder conflict scenario; the second option is concurrent, where the formulation takes into account the entire air traffic situation simultaneously and then computes an avoidance solution [3]. In

general, algorithms that use a pairwise formulation method will produce a sub-optimal avoidance solution whereas a concurrent formulation allows for an optimal solution in a multi-intruder environment. Figure 2.1 pictorially depicts both formulation methods. As an example, the TCAS algorithm employs a pairwise approach which can potentially fail in certain situations [3]. Kuchar and Yang [3] suggest that collision avoidance designers at a minimum examine their algorithms in a multiple intruder environment to determine robustness to potentially taxing multi-intruder scenarios especially for pairwise formulation methods. The algorithm presented in this body of work uses a concurrent formulation in determining an optimal collision avoidance solution for a multi-intruder environment.

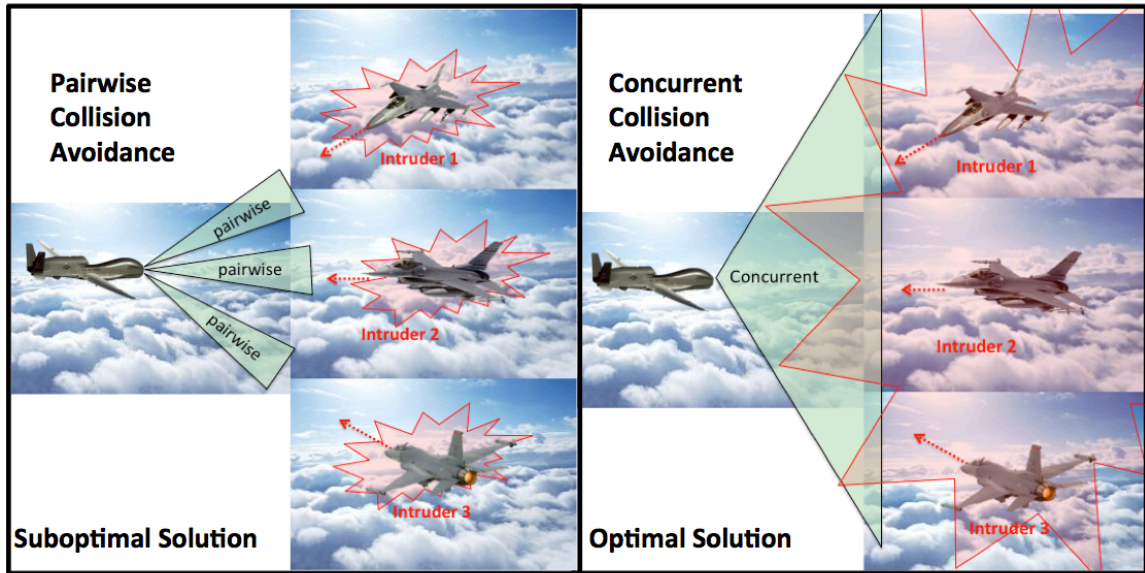


Figure 2.1: Pairwise Versus Concurrent Collision Avoidance Solutions. Adapted from [3]

2.1.2.2 Global vs. Local Avoidance.

By definition, a trajectory planning algorithm must produce a feasible path for an unmanned aircraft to fly that connects a start point to an end (or goal point) while avoiding known obstacles [25]. In this research, this type of trajectory planning algorithm is known as a *global* flight planning algorithm [26]. In general, these types of algorithms are used by mission planners to determine an *a priori* flight path for the RPA to fly while taking into consideration known obstacles or avoidance areas (such as no-fly zones, etc). There are

many algorithms that perform this task in the mission planning phase to determine a global flight path for the RPA to fly based on mission specific criteria. For instance, Mujumdar and Padhi [4] list the following four examples of global path planning algorithms: graph search methods [26], Rapidly-exploring Random Tree (RRT) [27], potential field [28], and a minimum effort guidance method [29].

Once airborne, if sensors onboard the RPA detect an airborne intruder or an unplanned obstacle, then a *local* collision avoidance algorithm must calculate a feasible (that is, flyable) path for the unmanned aircraft that avoids the threat. The focus of the work herein is not on global path planning algorithms but on local collision avoidance algorithms; nevertheless, researchers have adapted certain global planning algorithms, such as the four listed above, to also provide local avoidance solutions. However, not all of these algorithms are suitable for an airborne collision avoidance application. For example, Mujumdar and Padhi [25] note that the random nature associated with the RRT algorithm makes reliable collision avoidance with a moving intruder impossible. Therefore, RRT and other path planning algorithms, that despite reasonable adaptation, may still not provide reliable collision avoidance solutions were not considered in the literature review that follows.

In general, an RPA will have a start point and a flight plan endpoint or goal that the vehicle is attempting to achieve. After the local collision avoidance maneuver commands the RPA off the pre-computed flight path, there are two options the RPA can take: (1) compute a new optimal solution to reach the goal or (2) return to the original pre-computed flight path. Figures 2.2 and 2.3 pictorially depict both scenarios. The literature review considered both types of algorithms. The body of research presented herein can either calculate a new optimal solution to the goal or calculate a solution back to the original route of flight in an optimal manner.

2.1.2.3 Collision vs. Obstacle Avoidance.

Although not universally applied in the research community, to help distinguish between air and ground collision avoidance algorithms, the research herein adopts the nomenclature used by Wang et al. [30], where the phrase *collision avoidance* refers to

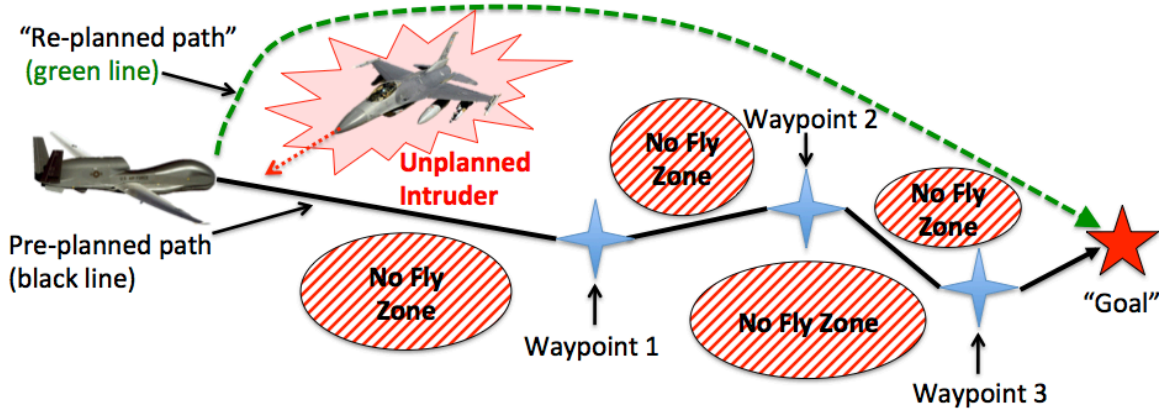


Figure 2.2: Replan by Calculating New Optimal Solution for the Goal. Adapted from [4]

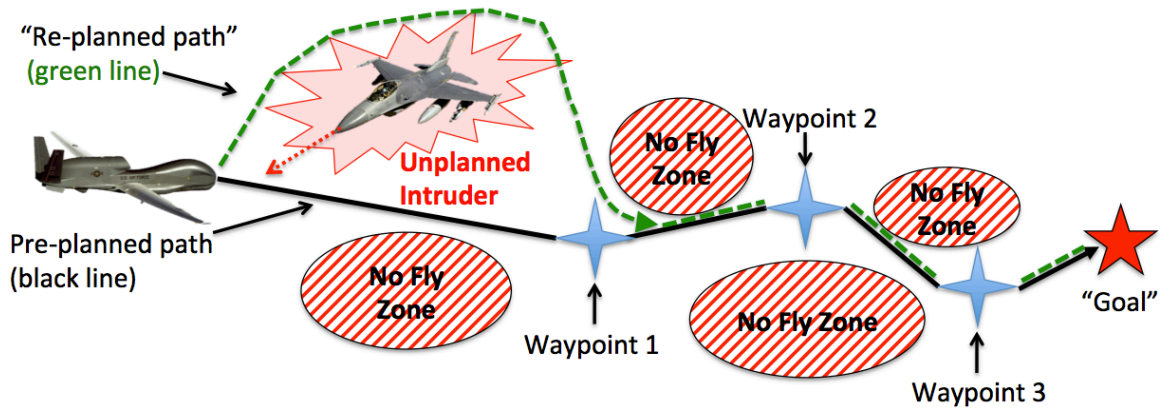


Figure 2.3: Replan by Calculating Optimal Solution Back to Path. Adapted from [4]

the RPA avoiding a moving airborne target and the phrase *obstacle avoidance* refers to the RPA avoiding a stationary or fixed obstacle. The focus of the work herein remains to develop best techniques for calculating optimal collision avoidance trajectories for an unmanned aircraft to avoid colliding with other airborne aircraft; however, the algorithm developed herein can also apply to fixed obstacles such as no-fly zones. A final note on terminology, in this document the term *ownership* refers to the unmanned aircraft (that is, the host RPA) and the term *intruder* refers to the target or *intruding aircraft*, which could be one or many.

2.2 Literature Review

This section provides a review of the literature in the following three key SAA topics: (1) collision avoidance algorithms, (2) stochastic estimation techniques for collision avoidance applications, and (3) dynamic models to characterize aircraft response.

2.2.1 Collision Avoidance.

A keyword search of “collision avoidance” in *ScienceDirect*, an online scholarly database, returned over 10,000 articles. Clearly, collision avoidance is a topic of great interest across a broad spectrum of modern transportation applications, and in particular, of grave interest to the aviation community. Since the 1950’s the engineering and aviation communities have sought technical solutions to prevent midair collisions between aircraft [31]. In 1955, the RTCA “working through the Air Transport Association (ATA), issued a request for the electronics industry to develop an Airborne Collision Avoidance System” [31]. A tragic midair collision the following year by two airliners over the Grand Canyon rapidly increased the urgency by the aviation and engineering communities as well as lawmakers to field a viable collision avoidance system. Over the next three decades, this ensuing national and international development effort formed the foundation for what is today TCAS, a limited *non-automated* collision avoidance system and the only collision avoidance system required onboard all large commercial and military transport aircraft.

Near the start of the twenty-first century, two key events contributed to an explosive growth in airborne collision avoidance research. The first was the desire to facilitate *free flight* by commercial aviation, and the second, addressed earlier, was the explosive growth of RPA’s. As part of the Next Generation Air Transportation System, free flight is a proposed ATC scheme that would increase airspace capacity by granting air traffic greater autonomy and the ability to bypass structured airways in order to maximize fuel efficiency or minimize time [32]. An obvious safety concern is this ATC scheme eliminates the inherent layer of protection against midair collisions imposed by a structured airway system. In an effort to facilitate free flight, a number of researchers posed the use of various airborne collision avoidance methods as cited by Christodoulou and Kodaxakis [33]. Similarly, in an effort

to increase safety and gain airspace access by RPAs, an avalanche of research in unmanned aircraft collision avoidance immediately followed on the heels of this spike in research to support free flight.

Useful surveys of collision avoidance algorithms appear in Kuchar and Yang [3], Goerzen et al. [34], and Mujumdar and Padhi [4]. In their seminal survey study, Kuchar and Yang reviewed nearly 70 conflict detection and resolution modeling methods and developed for the first time a taxonomy to classify these various modeling methods. In their survey, Mujumdar and Padhi specifically addressed evolving philosophies concerning collision avoidance methods for unmanned aircraft and defined two broad categories for RPA collision avoidance algorithms: global (primarily conducted off-line in mission planning) and local (conducted real-time while airborne). Goerzen et al. published a survey of motion planning algorithms for autonomous guidance of unmanned aircraft. In this survey the authors noted that the lack of exact algorithms or consensus on approximating algorithms unique to the RPA motion planning problem has made it difficult to design a guidance system, let alone choose an appropriate algorithm [34].

A number of works have looked at trajectory planning and optimization for air vehicles using an optimal control problem formulation methods [35–37] and some, such as [38], have even demonstrated this method in flight on a small-size unmanned vehicle. However, much of the previous work in trajectory optimization using optimal control problem formulation methods focused on deterministic path constraints such as fixed objects (buildings or no-fly zones) or avoiding aircraft with deterministic flight trajectories and fixed-uncertainty regions, and therefore, do not offer a robust treatment of handling time-varying uncertainty as required for the airborne collision avoidance problem. For example, [39] formulated the collision avoidance problem using a stochastic optimal control approach but limited their evaluation to atmospheric uncertainty against an intruder flying a known flight path. Other stochastic optimal control formulation methods such as generalized polynomial chaos (gPC) [40, 41] are computationally unwieldy and are not well-suited for real-time airborne collision avoidance.

An initial literature review revealed nearly 20 different problem formulation methods with potential applicability for an unmanned aircraft collision avoidance application. Even if the methodology presented by the researchers in the literature was not specifically devised for an RPA collision avoidance application (such as piloted *free flight* operations), this review still considered these methods since a logical extension such as an inner-loop controller could allow these methods to work for an RPA collision avoidance application. Likewise, although the focus of this review was on collision avoidance against a moving airborne intruder, this review still considered certain avoidance methods that researchers have implemented specifically for stationary obstacles. The logic for the inclusion of these methods is the same as previous; in certain cases, a logical extension could enable these methods to become applicable for a collision avoidance scenario. Appendix A to this document lists these various other problem formulation methods.

2.2.2 Stochastic Estimation.

Volumes of work in the literature address nonlinear estimation and the frequency of references to nonlinear filters continues to grow as researchers adapt and modify existing filters. Daum [42] provides an excellent survey of nonlinear estimation filters. In this review, Daum talks about the advantages and disadvantages of the various nonlinear filters. The number of specific navigation/tracking/avoidance algorithms in the literature is extremely vast. A few examples are in Jansson and Gustafsson [43] who implemented a nonlinear filter for a collision avoidance application for an automobile. Gustafsson et al. [44] used a nonlinear filter (particle filter) in a variety of tracking applications from a fighter aircraft using digital terrain to a car driving on the road.

For airborne collision avoidance a key consideration is estimating pilot intent [3, 45, 46] which largely contributes to uncertainty in an intruder's future position, and this uncertainty tends to grow as a function of time [47]. Researchers have used a number of different problem formulation methods for collision avoidance [3, 4, 34], but most of these methods do not offer a robust treatment of intruder uncertainty [3, 46]. The most common means to account for intruder uncertainty is to extrapolate a point estimate of the most likely trajectory (usually

based on a linear model) or use a worst-case dynamics trajectory over a finite-time horizon [3, 45, 46] as depicted in Figure 2.4 (adapted from [3]). Both extremes lead to suboptimal solutions by either underestimating an intruder’s state leading to a potential collision or overestimating the state leading to unnecessary trajectory deviations; however, true probabilistic approaches that take the “middle-ground” between these extremes are often avoided due to computational intractability [45]. Nonetheless, [46] showed that accounting for the full posterior distribution vice a most likely point estimate not only provides a more accurate estimate of intruder position, but also a more robust collision avoidance solution. The work herein uses a particle filter implementation referred to as sampling importance resampling (SIR) [6, 48] to estimate the intruder’s posterior distribution.

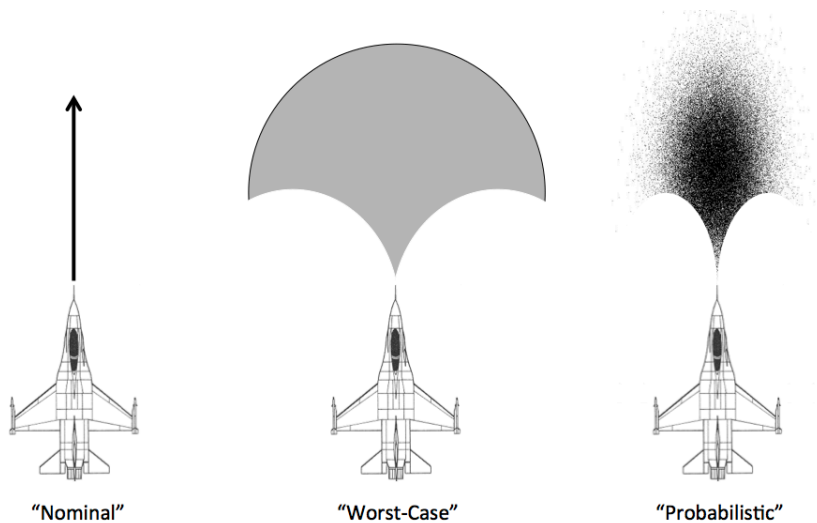


Figure 2.4: Intruder State Propagation Methods. Adapted from [3].

Particle filters have been used in the literature for collision avoidance in general [43] and for unmanned aircraft collision avoidance in particular [49, 50]. For instance, in [49] the authors used a particle filter to estimate uncertainty for a multiple unmanned vehicle environment; however, in this scenario each unmanned aircraft knew the planned trajectories of the other aircraft and the uncertainty was largely in atmospheric disturbances such as wind. Further, [50] used a particle filter with linear dynamics to assess collision

risk by estimating the distance at the closest point of approach; however, neither of these unmanned collision avoidance applications used the particle filter to estimate time-varying path constraints based on the posterior distribution calculated using nonlinear time-correlated dynamics. Others have used particle filters to expedite the belief state search [51] or to assess system performance [52] but not directly to determine the avoidance solution.

A large body of work exists which accounts for intruder uncertainty in a dynamic programming framework using a partially observable Markov decision process (POMDP) [46, 51, 53, 54]; however, a potential limitation with this approach is the ‘curse of dimensionality’ [46, 55], which can be intractable in a multiple intruder environment. Although recent efforts have used pairwise decomposition to handle multiple intruder avoidance in a dynamic programming framework [56], these solutions are not necessarily optimal. Furthermore, probabilistic approaches cited in the literature often limited the avoidance maneuver to a single 2D plane [39, 45, 46, 51, 52, 57].

2.2.3 Dynamic Models.

2.2.3.1 Intruder Model.

There are many sources in the literature that list dynamic models to characterize aircraft response. Several excellent resources are the classic text by Blackman and Popoli [58] and the survey series produced by Li and Jilkov [59, 60]. For collision avoidance applications, there are two categories for intruder motion models: maneuvering and non-maneuvering [59]. The MIAA program adopts a simple point-mass intruder dynamic model of 6 states (3 position states and 3 velocity states) [2]. This model assumes the intruder is effectively non-maneuvering (that is, flying straight and not turning); however, the model adds a process noise variance term into the estimation filter to help reduce tracking errors for a potentially maneuvering intruder, thus, “trading estimation accuracy for robustness.”

From the literature, another possible intruder model for a SAA application is the *Singer acceleration model* described in [58, 59, 61]. This is also a non-maneuvering model that represents intruder movement in three-dimensions where the states are position, velocity and acceleration. This model assumes the intruder’s acceleration $a(t)$ to be a zero-mean first-

order stationary Markov process with autocorrelation $R_a(\tau) = E[a(t+\tau)a(t)] = \sigma^2 \exp^{-\alpha|\tau|}$, or equivalently, power spectrum $S(\omega) = 2\alpha\sigma^2/(\omega^2 + \alpha^2)$ [59].

From Li and Jilkov [59], the state-space representation of the continuous-time Singer model is

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -\alpha \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} w(t) \quad (2.1)$$

and the equivalent discrete-time model is

$$x_{k+1} = F_\alpha x_k + w_k = \begin{bmatrix} 0 & T & (\alpha T - 1 + \exp^{-\alpha T})/\alpha^2 \\ 0 & 1 & (1 - \exp^{-\alpha T})/\alpha \\ 0 & 0 & \exp^{-\alpha T} \end{bmatrix} x_k + w_k \quad (2.2)$$

Li and Jilkov note “the Singer model relies on an accurate determination of the parameters α and σ^2 .” In this model, the parameter $\alpha = 1/\tau$ is the reciprocal of the maneuver time constant τ . For an aircraft application, a τ of approximately 60 seconds represents a “lazy turn” and a τ between 10 – 20 seconds represents a highly maneuvering target [59]. An additional benefits of the Singer model is that the formulation is extremely versatile in that (1) as τ increases the model reduces to the constant acceleration (CA) model and (2) as τ decreases the model reduces to the constant velocity (CV) model [59]. Tirri et al. [50] used a Singer model to represent intruder dynamics in their SAA application. However, the Singer acceleration model assumes that the aircraft’s x , y , and z positions operate independently, which is usually not the case for an aircraft. Thus, Blackman and Popoli [58] present other models such as the *Coordinated Turn Model* which is a maneuvering model that accounts for the natural correlation between the aircraft’s position states that occur during a coordinated turn maneuver. Both the Singer acceleration and the coordinated turn models are fully developed in the subsequent chapters of this document within the framework of a collision avoidance scenario.

2.2.3.2 Ownship Model.

For the ownship dynamics model, the body of work herein uses a 3 DOF nonlinear point-mass model as presented and developed in [2]. This model closely resembles the models in the literature for similar airborne collision avoidance applications such as by Raghunathan et al. [35] and Liu et al. [41]. Due to the large separation distances in the NAS (up to 5 nautical miles horizontally and 2,000 feet vertically), the 3 DOF point-mass model provides the right balance between performance and computational complexity when compared to a higher dimensional 6 DOF model [2]. This 3 DOF model appears as [2]:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{z}(t) \\ \dot{\gamma}(t) \\ \dot{\chi}(t) \end{bmatrix} = \begin{bmatrix} V \cos \gamma(t) \cos \chi(t) \\ V \cos \gamma(t) \sin \chi(t) \\ V \sin \gamma(t) \\ \frac{N_z g \cos \mu(t) - g \cos \gamma(t)}{V} \\ \frac{N_z g \sin \mu(t)}{V \cos \gamma(t)} \end{bmatrix} \quad (2.3)$$

where the states, $\mathbf{x}(t)$, consist of the cartesian directions (x, y, z) , flight path angle (γ) , and heading angle (χ) and the controls, $\mathbf{u}(t)$, are bank angle (μ) for horizontal control and normal acceleration (N_z) for longitudinal or z -axis control where N_z is defined in the velocity-axis frame [2]. The remaining variables in equation (2.3) are ground speed (V) and gravitational acceleration (g) .

The model in equation (2.3) assumes that the earth is flat and non-rotating, which are both reasonable assumptions given the low relative speed of the ownship compared to the earth's rotation and the short duration of the avoidance maneuver. The additional assumptions for this model are the ownship performs turns using bank angle alone such that the sideslip angle (β) is zero and the side forces are negligible, which are standard aircraft assumptions for this type of application. Figure 2.5 (adapted from [2]) shows the forces of weight (W) , lift (L) , thrust (T) , and drag (D) acting on the ownship. Finally, this model assumes that the flight control system keeps the vehicle speed constant throughout the avoidance maneuver. Again, based on the short duration of the avoidance trajectory (≈ 30 seconds) this is a reasonable assumption especially relative to the engine spool-up time for

a typical HALE platform. As a result, varying speed during the avoidance trajectory was not considered for this collision avoidance application.

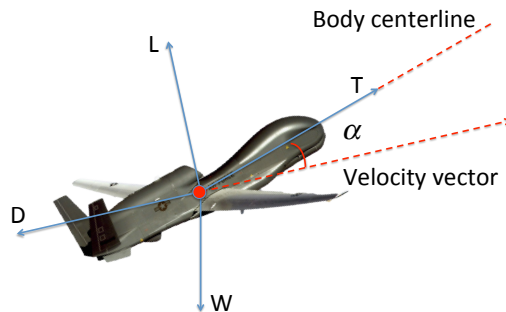


Figure 2.5: Forces Acting on the Aircraft. Adapted from [2].

Although point-mass models such as equation (2.3) are common in the literature, these types of models are only applicable for certain types of collision avoidance applications such as those where the minimum separation distance between aircraft is on the order of several hundred to several thousands of feet and the response times are on the order of tens of seconds to minutes. For example, in an air-to-air fighter aircraft *end-game* collision avoidance application such as described by Ikeda et al. [62] where the algorithm intentionally targets separation distances on the order of tens of feet and response times on the order of fractions of a second, a simple point-mass 3 DOF model would not be appropriate. A high-maneuvering dynamic application like that would require a higher fidelity and more accurate 6 DOF nonlinear aircraft model.

2.2.3.3 *Look-Ahead Time Horizon.*

Using the model in equation (2.3), the body of work herein applies a receding horizon approach and solves the optimal control problem over a fixed 30-second time horizon. This 30-second look-ahead provides an adequate response time for a large-size RPA to generate a well-clear separation distance without requiring aggressive maneuvers or inducing an

overtaxing computational burden. This 30-second look-ahead mirrors the look-ahead time for the MIAA program [2]. The system designers selected this 30-second time based on typical performance characteristics of a HALE unmanned aircraft. For example, a typical unmanned HALE aircraft will take approximately 15 seconds to generate a minimum of 500 feet separation distance and approximately 25 seconds to generate a “well-clear” separation distance, defined as 2,460 feet laterally and 820 feet vertically [2]. As a result, this 30-second look-ahead provides a reasonable balance between response time and computational efficiency.

2.3 Research Approach

The four distinguishing features of the work herein are: (1) the ability to perform local collision avoidance in three dimensions (3D) for a fixed-wing RPA both in a non-cooperative and cooperative environment while optimizing towards a global goal, (2) the ability to perform optimal concurrent collision avoidance vice sequential pairwise avoidance, (3) the ability to provide a discrete cost function to assess optimality against other optimal avoidance algorithms, and (4) the ability to account for uncertainty in the collision avoidance algorithm.

The work herein formulates the airborne collision avoidance problem as an optimal control problem and then numerically solve this problem by applying a pseudospectral method (aka the method of direct orthogonal collocation). As noted by Huntington [63], the terms *pseudospectral* and *direct orthogonal collocation* are essentially interchangeable. In recent years pseudospectral methods have been used in a wide variety of applications. For example, researchers have applied pseudospectral methods in areas ranging from minimizing propellant used by thrusters on the International Space Station [64] to calculating optimal basic fighter maneuvers (BFM) for a fighter aircraft in a dogfight [65]. The rapid advances in computing technology has made pseudospectral methods particularly appealing, especially in the field of autonomous motion planning for unmanned vehicles. In 2009, Gong et al. [64] presented an excellent review of pseudospectral motion planning applications for autonomous vehicles and showed “that it is possible to do motion planning for different problems under

the unified framework of optimal control and pseudospectral methods.” Betts [66] conducted a survey of numerical methods for trajectory optimization and the following references [67–69] performed surveys focused specifically on direct methods. The motivations for posing the collision avoidance problem for RPAs as an optimal control problem are: (1) this formulation is flexible and can easily incorporate time constraints, path deviations, and control use into an appropriate cost function to meet mission objectives while maintaining the required separation distance from one or more intruder aircraft; (2) these optimal solutions can provide design engineers a baseline or basis against which to judge other collision avoidance algorithms; and (3), design engineers can potentially use these optimal solutions in real-time RPA collision avoidance applications. Solving the collision avoidance optimal control problem requires knowledge of the intruder aircraft’s current and future position; however, in real-world applications the intruder(s) state information is unknown and must be estimated using imperfect sensor measurements and limited dynamic models. Therefore, reducing the error covariance and increasing the performance of the collision avoidance algorithm requires robust stochastic estimation techniques. The following chapters of this document provide additional details on the pseudospectral and stochastic estimation methods as they apply to the body of work herein.

III. Optimal Control Methodology

THIS CHAPTER describes the optimal control problem associated with airborne sense and avoid and uses a direct orthogonal collocation method to solve this problem. This chapter² then analyzes these results for different deterministic collision avoidance scenarios. This chapter contains five sections. Section 3.1 provides a review of the research methodology. Section 3.2 describes the optimal control problem associated with airborne collision avoidance and the direct orthogonal collocation method used to solve this problem. Section 3.3 develops the foundation for the work herein by demonstrating the research methodology using a simpler 2D deterministic model and describes different collision avoidance scenarios and associated cost functions and control penalties. Finally, Section 3.4 analyzes these 2D results for different collision avoidance scenarios and Section 3.5 summarizes these results and introduces the topic for the subsequent chapter on incorporating stochastic estimation into the optimal control problem.

3.1 Methodology

3.1.1 Direct Orthogonal Collocation.

The work herein applies direct orthogonal collocation to develop optimal collision avoidance trajectories. A number of doctoral dissertations published in the last decade provide a thorough treatment of this method for solving optimal control problems [40, 63, 71, 72]. In general, optimal control problems fall into two categories: indirect methods and direct methods [63]. In an indirect method, the researcher derives the first-order necessary conditions for optimality “via the calculus of variations and Pontryagin’s minimum principle” [63]. In a direct method the researcher transcribes the continuous-time optimal control problem “directly to a nonlinear programming problem (NLP) without formulating an alternate set of optimality conditions” [63]. The resulting NLP is then

²Note: This chapter appears as a conference paper with the co-authors listed in [70].

solved numerically using well-developed algorithms that attempt to satisfy the Karush-Kuhn-Tucker (KKT) conditions associated with the NLP [63].

3.1.2 Receding Horizon.

A fundamental concern in any airborne collision avoidance system is accounting for uncertainty in the intruder’s state information based on imperfect sensor measurements. To help mitigate and minimize the impacts of uncertainty for an airborne collision avoidance application researchers such as Rathbun et al. [73] and Frew et al. [74] utilized a receding horizon approach. This approach attempts to minimize uncertainty and facilitate real-time implementation by transitioning an open-loop control scheme into a closed-loop scheme through the use of a finite-time horizon. The method in the work herein likewise applies a receding horizon approach and solves the optimal control problem over a fixed 30-second time horizon.

Although the receding horizon approach does not implicitly account for uncertainty, this formulation is inherently resilient to uncertainty due to the iterative nature of the algorithm. In short, the receding horizon approach is an open-loop control approach specifically designed to provide appropriate control inputs for the vehicle (RPA) until the end of the set time horizon; however, in most real-time implementations the algorithm calculates and sends a new set of controls to the vehicle prior to the end of the finite horizon. Although system designers may use other problem formulation methods to generate the collision avoidance maneuver, designers often will employ a receding horizon approach to minimize the effects of uncertainty and to enable real-time implementation. For example, Rathbun et al. [73] primarily used a genetic problem formulation method but they also incorporated a receding horizon approach to account for uncertainty and generate a real-time trajectory. Similar to the approach described by Rathbun et al., the optimal control algorithm in the work herein plans and calculates the optimal set of control inputs over a fixed 30-second time interval, flies the first time step while planning and calculating again the next 30-second interval. This pattern is repeated which effectively transforms the optimization algorithm “from a static planner into a dynamic planner” [73] where

the previous control solution is the initial guess for the subsequent planning horizon. However, the performance of receding horizon algorithms can suffer in the absence of regular measurement updates since the planned trajectory may not accurately account for intruder state uncertainty throughout the finite time horizon.

3.1.3 Stochastic Estimation Method.

To account for intruder state uncertainty throughout the finite time horizon, the work herein applies a particle filter to estimate the intruder's current and future position as well as to characterize the three-dimensional (3D) uncertainty volume surrounding the intruder throughout the time horizon trajectory. Based on the results of the particle filter, the 3D uncertainty ellipsoid characterizing the intruder's current and future position expands and contracts as a function of the probability density function (PDF) associated with the observed states and the measurement update rate. The work herein uses the particle filter implementation referred to as Sequential Importance Resampling (SIR), which is a common implementation for similar tracking applications [6, 48, 50].

3.2 Optimal Control Problem Formulation

The formulation of the optimal control problem is to find an *admissible control* \mathbf{u} which causes the system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \quad (3.1)$$

to follow an *admissible trajectory* \mathbf{x} that minimizes the performance measure

$$J = \phi(\mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t) dt. \quad (3.2)$$

This results in an *optimal trajectory* \mathbf{x}^* and an *optimal control* \mathbf{u}^* where \mathbf{u}^* is defined as

$$\mathbf{u}^*(t) = \mathbf{a}(\mathbf{x}(t), t) \quad (3.3)$$

and \mathbf{a} is called the *optimal control law* [55]. In equation (3.2), L is the cost function associated with the trajectory and control usage costs, and ϕ is the cost function associated with the trajectory at time final, t_f .

In addition to the dynamic equality constraint equation (3.1), the system in equation (3.1) is subject to the following equality boundary and inequality path constraints:

$$\psi(\mathbf{x}(t_0), t_0, \mathbf{x}(t_f)) = 0 \quad (\text{equality boundary constraints}) \quad (3.4)$$

$$\mathbf{S}(\mathbf{x}(t)) \leq 0 \quad (\text{inequality path constraints}) \quad (3.5)$$

These constraints, equations (3.1, 3.4, and 3.5), bound or constrain the range of possible collision avoidance trajectories, $\mathbf{x}(t)$. In essence, the constraints are the non-negotiable system requirements the collision avoidance trajectory must satisfy. The performance measure, on the other hand, is the design parameter the optimal control problem seeks to minimize while satisfying the constraint requirements. This tradeoff between satisfying the system constraints while minimizing the performance measure is the heart of the optimal control problem. For the SAA problem, there are many performance measures or cost functions that the control designer can choose to minimize such as control usage, deviation from planned course, time traveled, and others. Each performance measure translates to a different physical characteristic for the RPA's optimal avoidance trajectory. The following five equations from [55, 75] list common performance measures along with a brief description of their physical significance. This chapter will demonstrate the results of minimizing various performance measures and highlight the differences as they relate to the SAA problem.

1. Minimize the operational time

$$J = (t_f - t_0) = \int_{t_0}^{t_f} 1 dt \quad (3.6)$$

2. Minimize the control effort

$$J = \frac{1}{2} \int_{t_0}^{t_f} (\mathbf{u}^T \mathbf{R} \mathbf{u}) dt, \quad \mathbf{R} > 0 \quad (3.7)$$

3. Minimize state deviations from a time varying path $\mathbf{C}(t)$ with minimum control effort

$$J = \frac{1}{2} \int_{t_0}^{t_f} [(\mathbf{x} - \mathbf{C})^T \mathbf{Q} (\mathbf{x} - \mathbf{C}) + (\mathbf{u}^T \mathbf{R} \mathbf{u})] dt, \quad \mathbf{Q} \geq 0 \quad \mathbf{R} > 0 \quad (3.8)$$

4. Minimize state deviations from the origin with minimum control effort

$$J = \frac{1}{2} \int_{t_0}^{t_f} [\mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{u}^T \mathbf{R} \mathbf{u}] dt, \quad \mathbf{Q} \geq 0, \quad \mathbf{R} > 0 \quad (3.9)$$

5. Minimize the control effort while the final state \mathbf{x}_f reaches close to a constant \mathbf{C}

$$J = \frac{1}{2} (\mathbf{x}_f - \mathbf{C})^T \mathbf{S}_f (\mathbf{x}_f - \mathbf{C}) + \frac{1}{2} \int_{t_0}^{t_f} (\mathbf{u}^T \mathbf{R} \mathbf{u}) dt, \quad \mathbf{S}_f \geq 0, \quad \mathbf{R} > 0 \quad (3.10)$$

The work herein uses a direct orthogonal collocation method to numerically solve the SAA optimal control problem. This approach transforms the dynamic optimization problem into a static optimization problem and then solves the static optimization problem using a direct orthogonal collocation method. In this method, “the state and control are approximated using global polynomials and collocation of the differential-algebraic equations is performed at orthogonal collocation points (i.e., the collocation points are the roots of an orthogonal polynomial and/or a linear combination of an orthogonal polynomial and its derivatives)” [76]. In the transcription process, the first step is to change the time interval of the optimal control problem from $t \in [t_0, t_f]$ to $\tau \in [-1, 1]$. This is done using the affine mapping [71]

$$\tau = \frac{2t}{t_f - t_0} - \frac{(t_f + t_0)}{(t_f - t_0)} \quad (3.11)$$

In general, this τ “mapping is used to replace the optimal control problem in equation (3.2) with the problem of minimizing” [71] the performance measure,

$$J = \phi(\mathbf{x}(1), t_f) + \frac{(t_f - t_0)}{2} \int_{-1}^1 L(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau \quad (3.12)$$

subject to the dynamic constraints

$$\frac{2}{(t_f - t_0)} \cdot \frac{d\mathbf{x}}{d\tau} = \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau \quad (3.13)$$

An advantage of a direct orthogonal collocation method, “that is, the *combination* of using global polynomials with orthogonally collocated points” is this method is “known to converge *spectrally* (i.e., converging to the solution faster than any power of N^{-m} where N is the number of collocation points and m is any finite value [Fornberg 1994])” [76]. An

often-stated critique of gradient-based NLP search methods is they produce local optimal solutions, which may or may not be global solutions. Subsequent chapters in this document address this limitation.

The previous chapter described the 3D ownship and intruder models for the predominance of the work herein. However, to first demonstrate the utility of the direct orthogonal collocation approach for an airborne collision avoidance application, this chapter first applies a simpler two-dimensional (2D), three-state deterministic ownship and intruder model to introduce the methodology and generate the results in Section 3.4.

3.3 Deterministic 2D Optimal Control Problem

We obtained the 2D deterministic results in this section by using the function *'fmincon'* in Matlab's[®] Optimization Toolbox with the Sequential Quadratic Programming (SQP) algorithm option selected. Applying a building block approach, this section begins with a single intruder aircraft and then progresses to multiple intruder aircraft scenarios. The simulation model in these scenarios assumes the *ownship*, which is the RPA, receives *perfect* measurement updates of the intruder aircraft current and future position(s). The next chapter removes this assumption and applies stochastic estimation techniques based on non-perfect sensor measurements. Like the 3 DOF model, in the 2D model the ownship maintains a constant velocity; however, the intruder aircraft can vary velocities. A 2D version of equation (2.3) for the ownship dynamics appear as the system dynamic constraints as shown in equation (3.14):

$$\begin{aligned}\dot{\mathbf{x}} &= V \cos(\chi) & \text{where } \mathbf{x} &= \text{Ownship's x-position (East)} \\ \dot{\mathbf{y}} &= V \sin(\chi) & \text{where } \mathbf{y} &= \text{Ownship's y-position (North)} \\ \dot{\chi} &= \mathbf{u} & \text{where } \chi &= \text{Ownship's heading (degrees)}\end{aligned}\tag{3.14}$$

and V is the ownship's ground speed.

3.3.1 Minimize Time.

There are numerous ways to setup the optimal control problem and the first set of scenarios investigated was to minimize the performance measure of time as shown in equation (3.6) with collision avoidance imposed as a path constraint. Minimizing this

performance measure simulates an RPA flying from one waypoint to the next waypoint in minimum amount of time. Operationally, this scenario can result from a tactical requirement to meet a minimum time-over-target-window. Therefore, for these scenarios the performance measure is simply:

$$\min J = (t_f - t_0) \quad (3.15)$$

The equality boundary constraints, equation (3.4), associated with the minimum time problem are initial time, initial positions for ownship and intruder aircraft, and the final position for the ownship. These boundary constraints appear as:

$$t_0 = \text{Initial time in seconds} \quad (3.16)$$

$$(x_0, y_0) = \text{Initial position (East, North) in feet for Ownship} \quad (3.17)$$

$$(x_{i_0}, y_{i_0}) = \text{Initial position (East, North) in feet for } i^{th} \text{ Intruder} \quad (3.18)$$

$$(x_F, y_F) = \text{Final position (East, North) in feet for Ownship} \quad (3.19)$$

The inequality path constraint, equation (3.5), for the collision avoidance problem is the ownship must maintain a minimum of 1,500 feet separation distance at all time from the intruder aircraft while seeking to fly to its specified final waypoint (x_F, y_F) in minimum time. Thus, the inequality constraint for this problem appears as:

$$\text{Minimum separation distance} \leq \sqrt{(\mathbf{x} - \mathbf{x}_i)^2 + (\mathbf{y} - \mathbf{y}_i)^2} \quad \forall t \quad (3.20)$$

For the Legendre-Gauss-Radau (LGR) direct orthogonal collocation method used, the collocation nodes include the interior points plus the initial point. The number of nodes are chosen to provide the desired solution fidelity. For this problem, we used Matlab[®] to generate both the τ points associated with the LGR collocation nodes and the differentiation matrix, \mathbf{D} . In addition, we used the algorithms developed by Shen et al.[77] to calculate the differentiation matrix \mathbf{D} as well as the quadrature weighting matrix \mathbf{W} in equation (3.25). By adding the final “+1” τ point to the user defined collocation nodes, the differentiation matrix (per design) is a non-square matrix. The advantage of using this form of the

differentiation matrix with the LGR method is this matrix allows the transcribed dynamic equations to conveniently propagate the aircraft state variables to their final value.

For the case where the performance measure is to “minimize time” to arrive at a fixed point while avoiding the intruder aircraft, the transcribed cost function appears as:

$$\text{Minimize: } J = \frac{(t_f - t_0)}{2} \int_{-1}^1 d\tau = (t_f - t_0) = t_f \quad (3.21)$$

Subject to the following transcribed dynamic constraints:

$$\dot{\mathbf{x}} : \mathbf{D} \cdot \mathbf{x} \cdot \frac{2}{(t_f - t_0)} = V \cos(\chi) \text{ where } \mathbf{x} = \text{Ownship's x-position (East)} \quad (3.22)$$

$$\dot{\mathbf{y}} : \mathbf{D} \cdot \mathbf{y} \cdot \frac{2}{(t_f - t_0)} = V \sin(\chi) \text{ where } \mathbf{y} = \text{Ownship's y-position (North)} \quad (3.23)$$

$$\dot{\chi} : \mathbf{D} \cdot \chi \cdot \frac{2}{(t_f - t_0)} = \mathbf{u} \quad \text{where } \chi = \text{Ownship's heading (degrees)} \quad (3.24)$$

3.3.2 Minimize Path Deviation.

For RPAs operating in the NAS, a more appropriate cost function is to minimize path deviation from the Air Traffic Control (ATC) pre-coordinated (or *sanitized*) air route or path. Minimizing deviations from this planned flight path allow FAA controllers to direct and schedule other air traffic around the RPA's proposed route. Further, in an operational environment the reconnaissance mission can dictate minimizing deviation from the planned path in order to maximize the RPA's surveillance potential. Equation (3.8) describes the performance measure associated with minimizing path deviations. Mathematically, this equation provides the tradeoff between minimizing path deviation (selecting \mathbf{Q}) and minimizing control usage (selecting \mathbf{R}). The relative ratio between these two competing weighting matrices is what shapes the optimal aircraft avoidance trajectory. In the 2D example, we chose the \mathbf{Q} -to- \mathbf{R} ratio to drive the RPA to quickly minimize path deviations but not so aggressively as to cause multiple path overshoots. Based on choosing diagonal matrices for \mathbf{Q} and \mathbf{R} , the transcribed cost function to minimize path deviation appears as:

$$\text{Minimize: } J = \frac{(t_f - t_0)}{2} \cdot \mathbf{e}^T \cdot \mathbf{W} \cdot \mathbf{Q} \cdot \mathbf{e} + \frac{(t_f - t_0)}{2} \cdot \mathbf{u}^T \cdot \mathbf{W} \cdot \mathbf{R} \cdot \mathbf{u} \quad (3.25)$$

where \mathbf{e} is the path deviation error and \mathbf{W} is the LGR quadrature weighting matrix.

An important point to note is that in order to use direct orthogonal collocation to solve the optimal SAA problem we transcribed the problem from the time domain to the τ domain using equation (3.11). This transformation requires us to map the cost (J) and the constraints, including the intruder(s) position, from the time domain to the τ domain using an affine transformation. Once transcribed, well-developed algorithms then solve the NLP numerically.

3.4 2D Results

This section provides the 2D results of the Matlab[®] simulations for the various optimal avoidance trajectory scenarios. For each of the simulations the blue aircraft represents the ownship, the red aircraft represents the first intruder aircraft, and the green aircraft represents the second intruder aircraft. The gray-colored line in the time sequence plots represent the ownship's intended flight path. In all simulation runs the ownship maintains a constant velocity of $V = 600$ feet/second (or 360 nautical miles/hour). Table 3.1 describes the various simulation scenarios. In this table the intruder aircraft velocities appear as a function of the ownship velocity in the east (x) and north (y) directions.

Table 3.1: Summary of Simulation Scenarios

Scenario	Intruder Dynamics	Performance Measure	Summary
Case 1	$v_{x1} = V \cdot \left(-\frac{1}{2}\right)$ $v_{y1} = V \cdot 0$	Minimize time to next waypoint	Nose-to-nose geometry where the Intruder flies a constant velocity of 300 feet/second.
Case 2	$v_{x1} = V \cdot \left(-\frac{1}{3}\right)$ $v_{y1} = V \cdot \left(\frac{1}{3}\right)$	Minimize time to next waypoint	Crossing geometry where the Intruder crosses the ownship intended flight path at a constant velocity.
Case 3	$v_{x1} = V \cdot -\cos\left(2\pi\frac{t}{18}\right)$ $v_{y1} = V \cdot \left(\frac{1}{2}\right)$	Minimize time to next waypoint	S-turn geometry where the intruder S-turns across the ownship intended flight path at a varying velocity.
Case 4	$v_{x1} = V \cdot \left(-\frac{1}{2}\right)$ $v_{y1} = V \cdot 0$	Minimize path deviation	Nose-to-nose geometry where the intruder “flies” a constant velocity of 300 feet/second .
Case 5	$v_{x1} = V \cdot \left(-\frac{1}{20}\right) t$ $v_{y1} = V \cdot 0$ $v_{x2} = V \cdot \left(-\frac{1}{2}\right)$ $v_{y2} = V \cdot \left(-\frac{1}{15}\right)$	Minimize path deviation	Intruder #1 accelerates toward the ownship while intruder #2 crosses the ownship’s intended flight path at a constant velocity.

3.4.1 Case 1: Single Intruder, “Nose-to-nose” Geometry, Minimize Time.

For this scenario, the ownship (blue) and intruder aircraft (red) started nose-to-nose at time initial ($t_0 = 0$ seconds) separated by one nautical mile (6,000 feet). For the simulation, the intruder aircraft (red) flew at a constant velocity equaled to one-half the ownship’s velocity along a straight line path towards the ownship’s starting position. In this scenario the ownship’s final waypoint position was located two nautical miles (12,000 feet) directly in front of the ownship’s starting position on the same latitudinal axis (East line). The simulation results appear in Figure 3.1 on the following page.

Figure 3.1 on the next page shows the optimal avoidance trajectory for this scenario was for the ownship (blue) to maneuver north in order to satisfy the inequality constraint of avoiding the intruder aircraft (red) by the specified separation distance of 1,500 feet. Once the optimal trajectory sat-

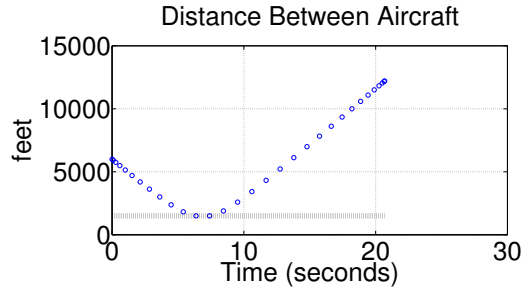


Figure 3.2: Case 1, Separation

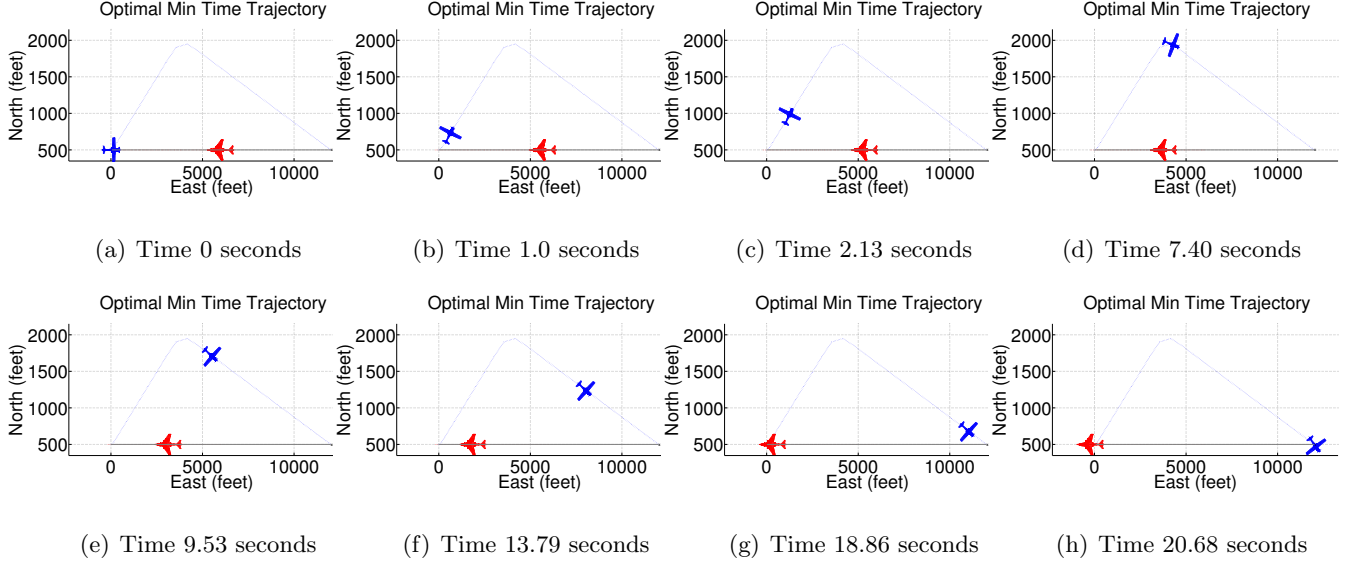


Figure 3.1: Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Intruder (Red) by 1,500 Feet Separation Distance While Minimizing Total Time.

ified the inequality path constraint, that is, the intruder aircraft (red) was no longer an active constraint, based on aircraft dynamics and boundary constraints the optimal path was then for the ownship (blue) to fly a straight-line path to the final waypoint in order to minimize time. Figure 3.2 on the preceding page shows the ownship and intruder aircraft separation distance as a function of time. The gray-colored dashed line in this figure indicates the 1,500 feet separation distance constraint. This figure confirms the optimal avoidance trajectory satisfied the inequality path constraint by maintaining at least 1,500 feet from the intruder aircraft.

3.4.2 Case 2: Single Intruder, “Crossing” Geometry, Minimize Time.

In this scenario, the simulation started with the ownship (blue) and intruder aircraft (red) separated by approximately one nautical mile and with the intruder aircraft’s flight path at a crossing angle to the ownship’s intended flight path. For this simulation the ownship’s final waypoint was located approximately 10,300 feet on a 15° angle from the ownship’s starting position. By design, the ownship started the simulation with a heading angle of 30° to simulate the RPA arriving at the current waypoint with a heading angle

not directly aligned to the next waypoint. The intruder aircraft (red) in this scenario flew a constant velocity flight path which intentionally intersected the ownship's intended flight path. The simulation results appear in Figure 3.3.

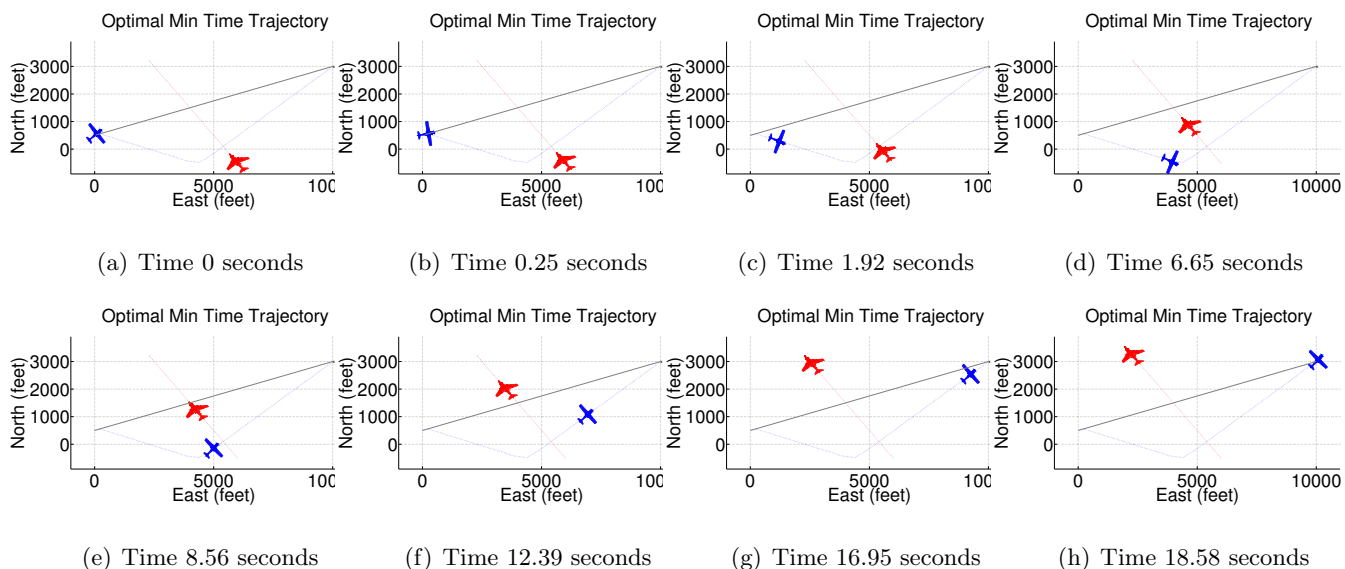


Figure 3.3: Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Intruder (Red) by 1,500 Feet Separation Distance While Minimizing Total Time.

Intuitively, the time sequence plots in Figure 3.3 show the optimization algorithm performed correctly in this scenario. In the simulation, the optimal avoidance trajectory was for the ownship (blue) to intentionally maneuver to the south to avoid the intruder aircraft's (red) flight path, which

crossed the ownship's intended flight path from southeast to northwest. As in Case 1, once the optimal trajectory satisfied the inequality path constraint, that is, the intruder aircraft (red) was no longer an active constraint the optimal trajectory was then for the ownship (blue) to fly a straight-line path to the final waypoint in order to minimize time. Figure 3.4 plots the aircraft separation distance as a function of time and confirms the optimal avoid-

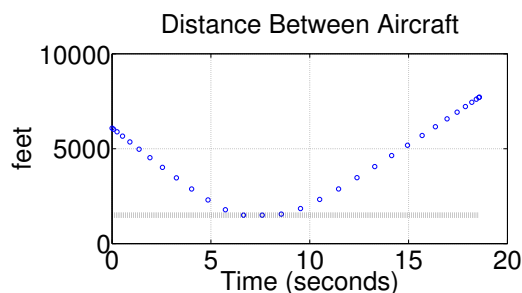


Figure 3.4: Case 2, Separation

ance trajectory satisfied the inequality path constraint by maintaining at least 1,500 feet from the intruder aircraft while minimizing the overall time to reach the final waypoint.

3.4.3 Case 3: Single Intruder, “S-Turn” Geometry, Minimize Time.

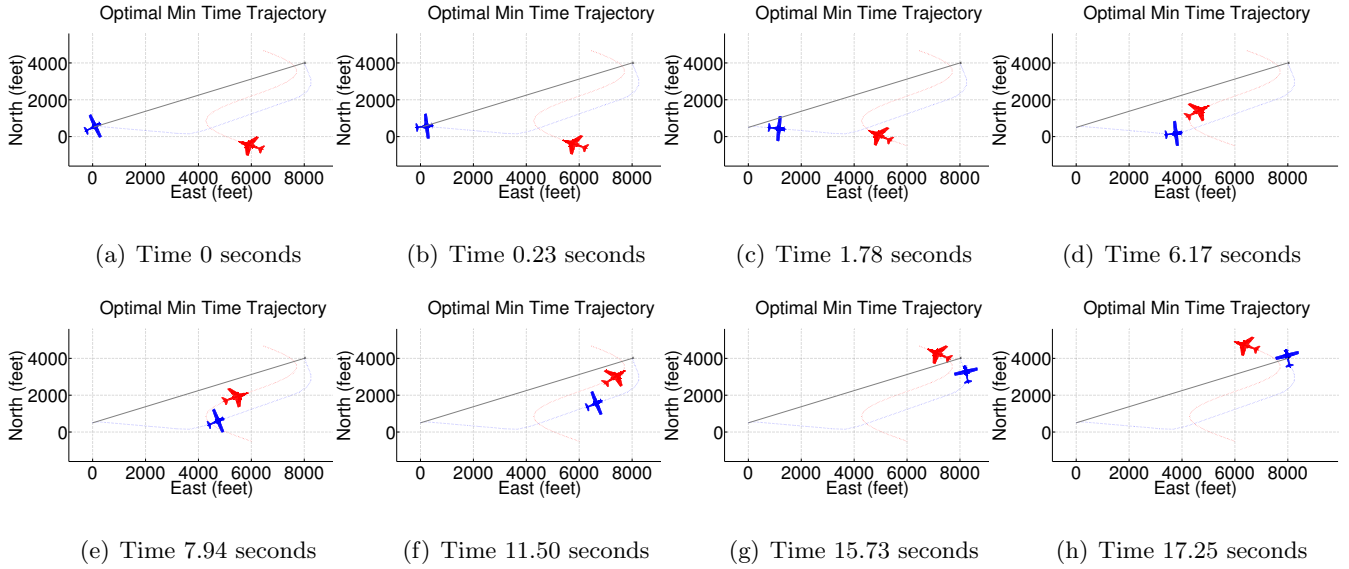


Figure 3.5: Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Intruder Aircraft (Red) by 1,500 Feet Separation Distance While Minimizing Total Time.

The initial setup for the ownship and intruder aircraft in this scenario was identical to that of Case 2. The difference in this scenario from Case 2 was the intruder aircraft (red) now flew a changing velocity S-Turn pattern that crossed the ownship’s intended flight path. This scenario intentionally challenged the robustness of the optimization algorithm. The simulation results appear in Figure 3.5.

Even with the challenging dynamics of a constantly turning intruder aircraft, the route planner performed correctly. The optimal minimum time avoidance trajectory was for the ownship (blue) to initially maneuver south to avoid the intruder aircraft (red). Once the intruder aircraft’s (red) trajectory S-turned and reversed directions, in

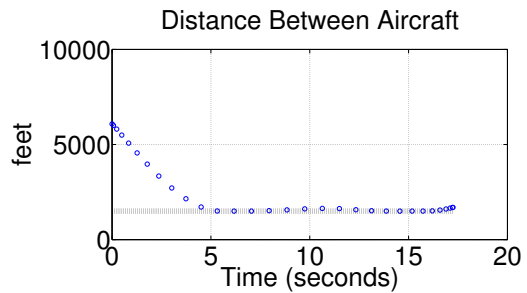


Figure 3.6: Case 3, Separation Distance

order to minimize time, the optimal path, as depicted in Figure 3.5 on the previous page, was for the ownship to maintain a position 1,500 feet behind the intruder aircraft enroute to the final waypoint. Figure 3.6 on the preceding page shows the separation distance as a function of time and confirms the optimal avoidance trajectory satisfied the inequality path constraint. Since there was no constraint on the final heading, the ownship arrived at the final waypoint with a heading angle not aligned with its initial flight path as shown in panel (h) of Figure 3.5 on the previous page; however, this arbitrary heading angle may be operationally undesirable. Therefore, to fix this problem we can either add a heading constraint at t_f or a path deviation constraint as demonstrated in the next case.

3.4.4 Case 4: Single Intruder, “Nose-to-nose” Geometry, Minimize Path Deviation.

The initial setup for the ownship and intruder aircraft in this scenario was identical to that of Case 1. The difference in this scenario from Case 1 was the performance measure (J) now minimized path deviation along the intended route of flight instead of time by using equation (3.25). The simulation results appear in Figure 3.7 on the following page. Like Case 1, the ownship (blue) and intruder aircraft (red) started nose-to-nose at time initial ($t_0 = 0$ seconds) separated by one nautical mile (6,000 feet). The intruder aircraft (red) flew at a constant velocity equal to one-half the ownship’s velocity along a straight line path towards the ownship’s starting position. In this scenario the ownship’s intended flight path was due east along on the same latitudinal axis (East line) as its starting position. The time sequence plots in Figure 3.7 on the next page align with intuition and show the optimization algorithm performed correctly in this scenario. Like Case 1, the optimal avoidance trajectory for this scenario was for the ownship (blue) to maneuver north in order to satisfy the inequality constraint of avoiding the intruder aircraft (red) by the specified separation distance of 1,500 feet. However, unlike Case 1, once the optimal trajectory satisfied the inequality path constraint, that is, the constraint associated with the intruder aircraft (red) was no longer active, based on the new performance measure of minimizing

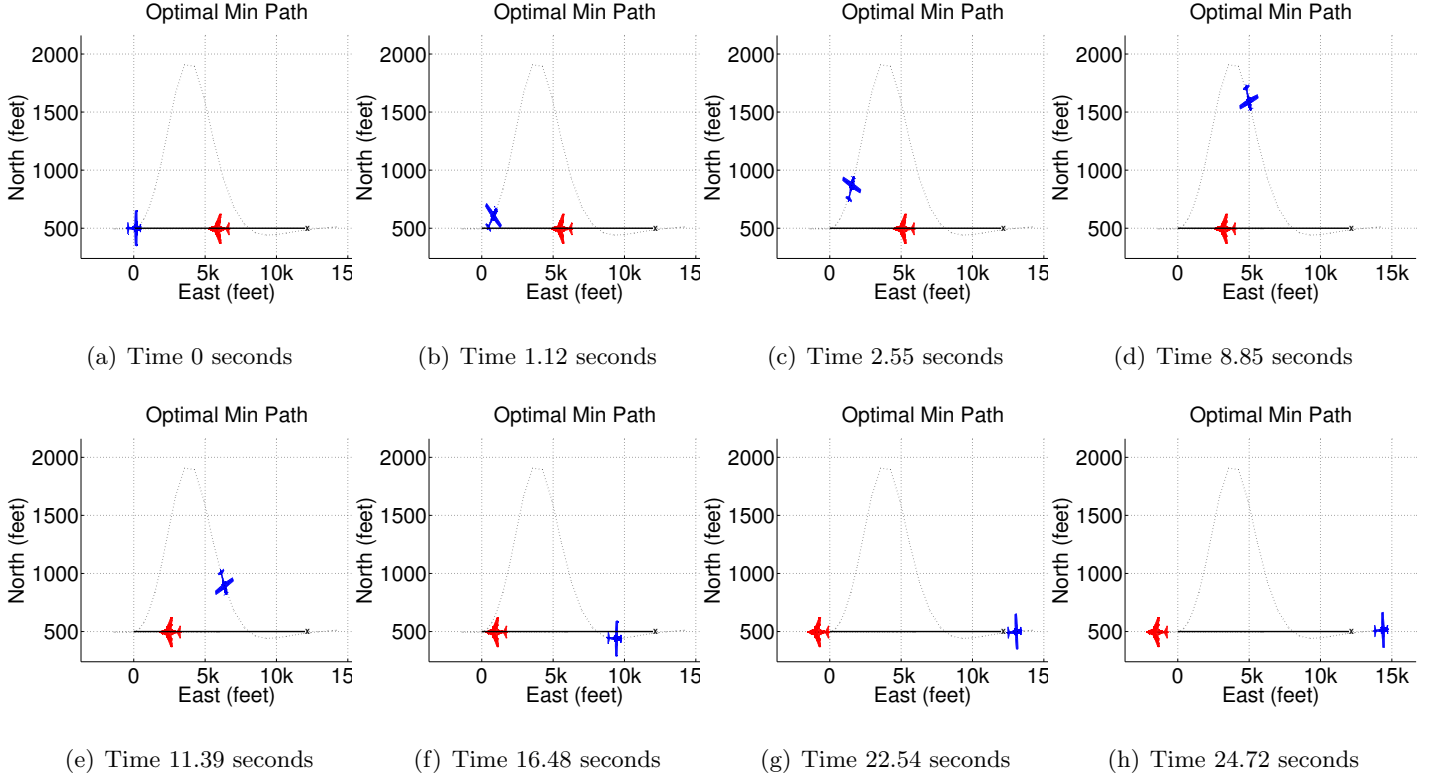


Figure 3.7: Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Intruder Aircraft (Red) by 1,500 Feet Separation Distance While Minimizing Path Deviation.

path deviations, the optimal path was now for the ownship (blue) to turn south and correct back to the intended flight path in order to minimize overall path deviation.

Panel (f) in Figure 3.7 shows the ownship experienced a slight overshoot then corrected back to the intended flight path line. This overshoot resulted from the **Q-to-R** ratio used to shape the optimal trajectory to quickly minimize path deviations but not so aggressively as to cause multiple path overshoots. When comparing this scenario to Case 1, the impact of the cost function on the optimal avoidance trajectory becomes very clear. In the minimum time scenario (Case 1), the optimal path was to fly a straight line

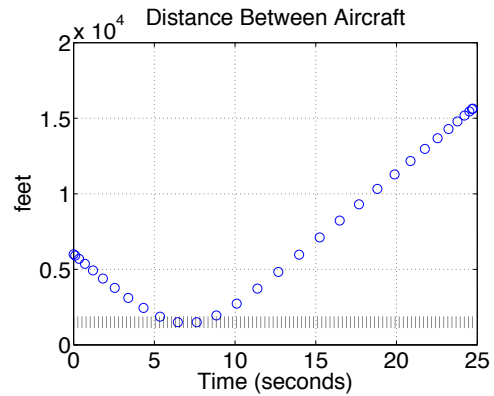


Figure 3.8: Case 4, Separation

path to the final waypoint once the path constraint associated with intruder aircraft was no longer active. In this minimize path deviation scenario of Case 4, the optimal trajectory was for the ownship to correct back to path once the constraint associated with the intruder aircraft was no longer active. Figure 3.8 on the previous page shows the separation distance as a function of time and confirms the optimal avoidance trajectory satisfied the inequality path constraint.

3.4.5 Case 5: Two Intruder, Accelerating “Nose-to-nose” and “Crossing” Geometry, Minimize Path Deviation.

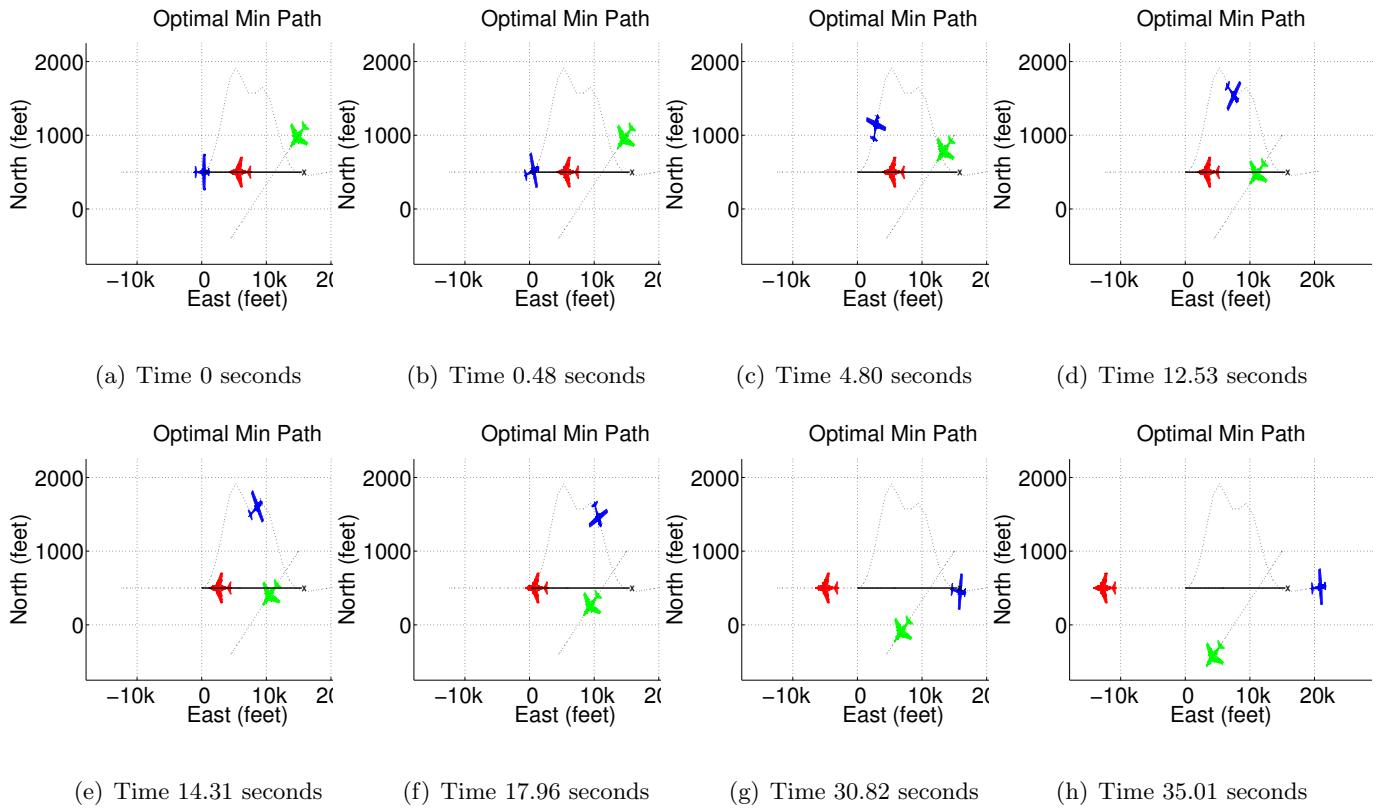


Figure 3.9: Time Series of Optimal Trajectory for Ownship (Blue) Avoiding Both Intruder Aircraft #1 (Red) and Intruder Aircraft #2 (Green) by 1,500 Feet Separation Distance While Minimizing Path Deviation.

This scenario involved two intruder aircraft. The first intruder aircraft (red) started nose-to-nose at one nautical mile from the ownship (blue) and then accelerated directly towards the ownship's starting point. The second intruder aircraft (green)

started in the northeast and then flew at a constant velocity in a southwesterly direction that crossed the ownship's intended flight path. The simulation results appear in Figure 3.9 on the preceding page. In this scenario, the optimal path was for the ownship (blue) to maneuver to the north to satisfy the active inequality path constraint associated with the first intruder aircraft (red) which accelerated towards the ownship's initial position. Unlike previous scenarios, due to the red intruder's slight acceleration, the separation distance from the ownship increased at an exponential rate.

On the other hand, like Case 4, once the optimal trajectory satisfied the inequality path constraint, that is, the constraint associated with the first intruder aircraft (red) was no longer active, based on the performance measure of minimizing path deviations, the optimal path was again for the ownship (blue) to turn south and correct

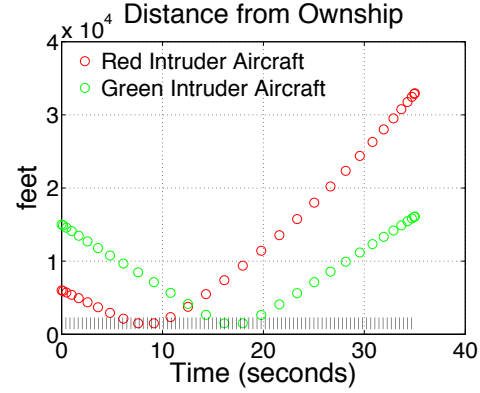


Figure 3.10: Case 5, Separation

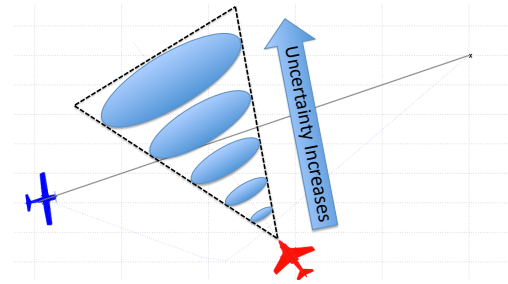
back to the intended flight path. However, at that time the second intruder (green) aircraft's flight path crossed the ownship's route of flight. Therefore, the optimal path was now for the ownship to turn momentarily to the north then resume back south in order to minimize the overall flight path deviation. Again, based on the **Q**-to-**R** ratio, the ownship (blue) slightly overshot and then corrected back to the intended flight path line as shown in panel (g) of Figure 3.9 on the previous page. Figure 3.10 shows the separation distance for both intruder aircraft as a function of time and confirms the optimal avoidance trajectory satisfied both inequality path constraints.

3.5 Summary of Optimal Control Results

The previous sections demonstrated the robustness of the optimal controller in satisfying the constraints and minimizing the specified performance measure. In each scenario, the route planner correctly avoided the intruder aircraft by the specified separation distance of 1,500 feet while minimizing the appropriate performance measure. For these

scenarios, the model operated on perfect or deterministic knowledge of the intruder aircraft current and future position(s). In real-world applications perfect knowledge is rarely, if ever, the case. Therefore, for a real-time airborne application, the route planner requires estimates from sensors such as an airborne radar, electro-optical sensors, or a cooperative datalink to provide a measurement or estimate of the intruder aircraft position(s); however, these measurements are not perfect nor are they always available.

Thus, the route planner requires a stochastic estimator to account for the uncertainties associated with the sensors and the uncertainties associated with propagating these measurements forward in time in order to produce a predicted intruder aircraft



flight path. As depicted in Figure 3.11, the uncertainty associated with the intruder aircraft grows as a function of time. To account for these uncertainties and provide the route planner with the “best” estimate of the intruder aircrafts’ current and future position, the next chapter evaluates the use of a particle filter for the optimal collision avoidance application. Using a stochastic rather than a deterministic approach (as in Section 3.3), the separation distance inequality constraint associated with the intruder aircrafts’ position must “expand and contract” as a function of time and measurement updates. The route planner must now satisfy a time varying inequality constraint based on the covariance of the estimates of the intruder aircrafts’ current and future position(s). A description of the particle filter algorithm follows in the next chapter as well as specifics of how to integrate the filter with the previously developed optimal route planner. The performance of the route planner using a stochastically driven model rather than a deterministic case will show the advantage of using estimation techniques in these scenarios.

IV. Estimation Methodology

THIS CHAPTER builds on the development from the previous chapter and establishes a framework for incorporating stochastic estimation into the airborne sense and avoid optimal control problem. This chapter introduces and describes the specifics of how to integrate a particle filter with the previously developed optimal route planner using a 3D stochastic collision avoidance scenario. This chapter³ consists of five sections. Section 4.1 describes the 3D models and Section 4.2 describes the particle filter implementation. Section 4.3 discusses the propagation and observation models used in the scenario and Section 5.6 analyzes the results. Finally, Section 4.5 concludes by introducing the follow-on research for the next chapter.

4.1 3D Model Description

The 3 DOF ownship model for this chapter is described by equation (2.3) with the assumptions for this model described in Section 2.2.3.2. To represent the intruder dynamics in 3D (x , y , and z), this chapter uses a nine-state Singer acceleration model [61] where the states are position, velocity and acceleration. As briefly described in Section 2.2.3.1, this model assumes the intruder's acceleration $a(t)$ to be a zero-mean first-order stationary Markov process with autocorrelation $R_a(\tau) = E[a(t+\tau)a(t)] = \sigma^2 \exp^{-\alpha|\tau|}$, or equivalently, power spectrum $S(\omega) = 2\alpha\sigma^2/(\omega^2 + \alpha^2)$ [59].

As shown in equation (2.1) and repeated here for convenience, the state-space representation of the continuous-time Singer model for one dimension is [59]

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -\alpha \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} w(t) \quad (4.1)$$

³Note: This chapter appears as a conference paper with the co-authors listed in [70].

where $\mathbf{x}(t)$ is a column vector of position, velocity, and acceleration. Likewise, the equivalent discrete-time model is [59]

$$\mathbf{x}_{k+1} = \mathbf{F}_\alpha \mathbf{x}_k + \mathbf{w}_k = \begin{bmatrix} 1 & T & (\alpha T - 1 + \exp^{-\alpha T})/\alpha^2 \\ 0 & 1 & (1 - \exp^{-\alpha T})/\alpha \\ 0 & 0 & \exp^{-\alpha T} \end{bmatrix} \mathbf{x}_k + \mathbf{w}_k \quad (4.2)$$

In this model, the parameter $\alpha = 1/\tau_m$ is the reciprocal of the maneuver time constant τ_m . For an aircraft application, a τ_m of approximately 60 seconds represents a “lazy turn” and a τ_m between 10 – 20 seconds represents a highly maneuvering target [59]. T represents the sample rate of the trajectory. In the case where $\alpha T \ll 1/2$, the discrete time propagation model and noise strength, \mathbf{Q}_k , become [61]

$$\mathbf{F} = \begin{bmatrix} T^5/20 & T^4/8 & T^3/6 \\ T^4/8 & T^3/3 & T^2/2 \\ T^3/6 & T^2/2 & T \end{bmatrix} \quad (4.3)$$

$$\mathbf{Q}_k = 2\alpha\sigma_m^2 \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \quad (4.4)$$

Li and Jilkov note “the Singer model relies on an accurate determination of the parameters α and σ^2 .” A benefit of the Singer model is that the formulation is extremely versatile in that: (1) as τ_m increases, the model reduces to the constant acceleration model and (2) as τ_m decreases, the model reduces to the constant velocity model [59]. Tirri et al. [50] used a Singer model with their particle filter to represent intruder dynamics in their SAA application.

4.2 Particle Filter Development

This research uses SIR for the particle filter implementation which is common in tracking applications of this type. This section describes the particle filter algorithm as defined in [6, 48]. The inputs to the particle filter algorithm are: $\boldsymbol{\chi}_{k-1}^{(+)}$, \mathbf{u}_k , \mathbf{z}_k , where $\boldsymbol{\chi}_{k-1}^{(+)}$ represents the particles at time $k - 1$ whose distribution defines the probability density

function (PDF) of the system state, \mathbf{u}_k is the control input of the system on the interval up to time k , and \mathbf{z}_k is the measurement at time k .

The algorithm first initializes the set of before-measurement particles, $\boldsymbol{\chi}_k^{(-)}$, and the set of after-measurement particles, $\boldsymbol{\chi}_k^{(+)}$, to matrices of all zeros [48].

$$\boldsymbol{\chi}_k^{(-)} = \boldsymbol{\chi}_k^{(+)} = \mathbf{0}$$

The filter uses the dynamic model which is a function of the particle's state information, the control input, and the process noise to propagate each of the N particles forward in time from $k - 1$ to k . Each particle also has an associated weight, w_k^i , describing the likelihood of the particle generating a measurement similar to the actual measurement. When normalized, the set of particle weights represent the PDF of the true state of the system. The weights are assigned as the likelihood of the particle given a measured value of the state at time k . The newly generated particles and weights are then added to the initial (zero) values of the particles, $\boldsymbol{\chi}_k^{(-)}$. These steps are described algorithmically as [48]

for $i = 1$ to N (where $N = \text{number of particles}$)

sample $\hat{\mathbf{x}}_k^i \sim p(\hat{\mathbf{x}}_k \mid \hat{\mathbf{x}}_{k-1}, \mathbf{u}_k)$

$w_k^i = p(\hat{\mathbf{z}}_k \mid \hat{\mathbf{x}}_k^i)$

$\boldsymbol{\chi}_k^{(-)} = \boldsymbol{\chi}_k^{(-)} + \langle \hat{\mathbf{x}}_k^i, w_k^i \rangle$

end

After the measurement, the particle weights are normalized by their sum, W , such that the weights now define the posterior PDF of the system, described algorithmically as [48]

calculate total weight $W = \sum_{i=1}^N w_k^i$

for $i = 1$ to N

normalize $w_k^i := w_k^i / W$

end

Given this new posterior distribution of particles, some particles may not represent the system state appropriately due to a very small weight. Therefore, the method in this chapter resamples all the particles from the posterior PDF. This method inherently emphasizes particles with higher value weights and reduces or removes particles with lower value weights. Finally, the algorithm adds the resampled particles to the set of “after measurement” particles, $\chi_k^{(+)}$ as follows [48]

```

for  $i = 1$  to  $N$ 
    draw  $j$  with probability  $\propto w_k^i$ 
    add  $\hat{\mathbf{x}}_k^j$  to  $\chi_k^{(+)}$ 
end
return  $(\chi_k^{(+)})$ 

```

The particle filter algorithm then iterates for the number of desired time samples as defined by the optimal control problem. Common estimation methods often include the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF). The advantages of the particle filter over the EKF and UKF are that the particle distribution and the measurement likelihood do not need to be Gaussian. The particles and their weights shape the PDF of the state whereas the EKF and UKF rely on solely propagating variants of a mean and standard deviation of the state. However, the disadvantage of the particle filter is the increased computational cost associated with propagating a large number of particles. Chapter VIII in this document analyzes the simulation results of the EKF, UKF, and particle filter and compares their performance for use in an airborne sense and avoid application with the NAS. The next section describes how the particle filter affects the 3D airborne collision avoidance scenario. Specifically, this section explains how the particle filter impacts the inequality path constraint volume or keep-out zone as well as the intruder’s estimated trajectory.

4.3 3D Scenario Description and Implementation

Section 3.3 referenced a 2D model for ownship and intruder dynamics as well as the optimization parameters used in the 2D case; however, unlike the 2D model, a 3D model

more accurately represents a real-world collision avoidance encounter. Thus, to model real world aircraft performance as well as estimate intruder uncertainty, the 3D collision avoidance scenario requires a different dynamics model for the intruder than previously described in the 2D deterministic scenarios. As described in Section 4.1, the intruder model we used for this scenario is the Singer acceleration model. In addition to a new dynamics model, the optimization problem is slightly different than previously described since the algorithm now continuously calculates a solution over a finite time horizon.

4.3.1 Scenario Description.

The application for this collision avoidance algorithm is for NAS flight operations where aircraft usually do not perform aggressive maneuvers. Therefore, consistent with the Singer acceleration model described in Sections 2.2.3.1 and 4.1, in this scenario the algorithm uses a τ_m of 60 seconds for the intruder to model a “lazy turn” with a sample rate of 1 Hz ($T = 1$ second). Since αT is sufficiently small we can use the approximation values for the Singer model expressed in equations (4.3) - (5.18). In the stochastic 3D scenario, the optimization algorithm attempts to minimize the deviation from a 3D flightpath corridor using maximum control available (bounded in the constraint function only). This generates an optimal 3D path for the ownship to fly that satisfies the boundary, path, and dynamic constraints. The simulation flies the ownship and intruder along the optimized trajectory and the truth trajectory, respectively, until the next user-defined measurement time (1 Hz).

4.3.2 State Propagation.

As described earlier, the optimization algorithm uses a fixed 30 second receding horizon trajectory. To meet this requirement, the particle filter generates 1,000 particles from an initial Gaussian distribution around a nominal nine-state vector representing position, velocity, and acceleration in 3D. For the collision avoidance scenario in this section the initial state mean and standard deviation values are given in Table 4.1.

Table 4.1: Initial Particle Value Distribution

State	Mean (feet)	Standard Deviation (feet)
x	9,000	0
y	453.7	0
z	7,150	0
v_x	200	0
v_y	20	0
v_z	0	0
a_x	0	0.81
a_y	0	0.26
a_z	0	0.06

After an initial draw from the distributions, the algorithm propagates each particle according to the dynamics of the Singer model for 30 seconds. The algorithm then arbitrarily selects one of the 1,000 particle trajectories as the “true intruder trajectory” in order to generate measurements and evaluate the model performance. The model calculates a mean and standard deviation of the particle distribution at each time step. The optimization then uses the mean value of the 3D position at each time step as the expected trajectory of the intruder and the standard deviation in each axis, σ_i , to define the minimum separation region or keep-out zone around that mean. This keep-out zone is an ellipsoid that varies as a function of the standard deviation of the position estimate in each axis. This ellipsoid becomes the inequality path constraint for the optimization. For this chapter, we adapt an ellipsoid with a minimum 1,000 feet radius in each dimension by adding one standard deviation of the position estimate in each axis. As a result, this ellipsoid both grows and contracts as a function of time and measurement update such that as the position estimate uncertainty increases the minimum separation region grows. Likewise, this region contracts after the measurement update as the position uncertainty decreases. We used GPOPS III [78] as the optimization software for this section.

4.3.3 Observation Model.

The sensor measurements at time k , \mathbf{z}_k , of the intruder aircraft are nonlinear functions of the intruder’s state, $\mathbf{h}(\mathbf{x}_k)$, modeled according to

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k) + \boldsymbol{\nu}_k \quad (4.5)$$

where $\boldsymbol{\nu}$ is zero-mean, white, Gaussian noise with noise strength,

$$\mathbf{E}[\boldsymbol{\nu}_j^T \boldsymbol{\nu}_k] = \mathbf{R}_m \delta_{jk} \quad (4.6)$$

where δ_{jk} is the Kronecker delta function. The noise autocorrelation, \mathbf{R}_m , indicates the uncertainty associated with the sensor. The noise strength associated with each measurement appears along the diagonal elements of the matrix \mathbf{R}_m . The simulation described in this chapter used the following noise autocorrelation matrix.

$$\mathbf{R}_m = \begin{bmatrix} (0.01)z_R \text{ ft}^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 \text{ deg}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 \text{ deg}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & (0.05)z_{v_x} (\text{ft/s})^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & (0.1)z_{v_y} (\text{ft/s})^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 (\text{ft/s})^2 \end{bmatrix} \quad (4.7)$$

These noise values are notional; however, the nonlinear filter evaluation in Chapter VIII uses the actual noise values from the MIAA program. At each measurement time, the algorithm performs slant range (z_R), azimuth (z_{az}), and elevation angle (z_{el}) measurements of the intruder by simulating a radar system onboard the ownship. In addition, the algorithm also performs a velocity measurement by simulating an ADS-B velocity input with a specified noise strength. The measurement equations appear as [2]:

$$\mathbf{h}(\mathbf{x}) = \begin{bmatrix} z_R \\ z_{az} \\ z_{el} \\ z_{v_x} \\ z_{v_y} \\ z_{v_z} \end{bmatrix} = \begin{bmatrix} \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \\ \tan^{-1} \frac{\Delta y}{\Delta x} \\ \sin^{-1} \frac{\Delta z}{\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}} \\ v_x \\ v_y \\ v_z \end{bmatrix} \quad (4.8)$$

We simulate an actual measurement by adding noise according to equation (4.5) using the truth trajectory. The particle filter algorithm uses equation (4.9) to update each particle

at the measurement time k .

$$\hat{\mathbf{z}}_k^i = \mathbf{h}(\mathbf{x}_k^i) \quad (4.9)$$

The residual, r , in equation (4.10) then determines a new particle weight, w , from the likelihood function in equation (4.11),

$$\mathbf{r}_k^i = \mathbf{z}_k - \hat{\mathbf{z}}_k^i \quad (4.10)$$

$$w_k^i \propto \exp\left(-\frac{1}{2}\mathbf{r}_k^T \mathbf{R}_m^{-1} \mathbf{r}_k\right) \quad (4.11)$$

A large residual results in a smaller weight, likewise a small residual results in a larger weight. Particles with smaller weights have a lower probability of being resampled, whereas particles with higher weights have a higher probability of being resampled. The updated and resampled particle distribution now define a new intruder initial condition and uncertainty volume. The particle filter again propagates the new resampled particles using the Singer acceleration model for a new fixed 30 second time horizon beginning at time k . The optimal control algorithm then uses this new estimated intruder trajectory and position uncertainty to determine a new ownship optimal path.

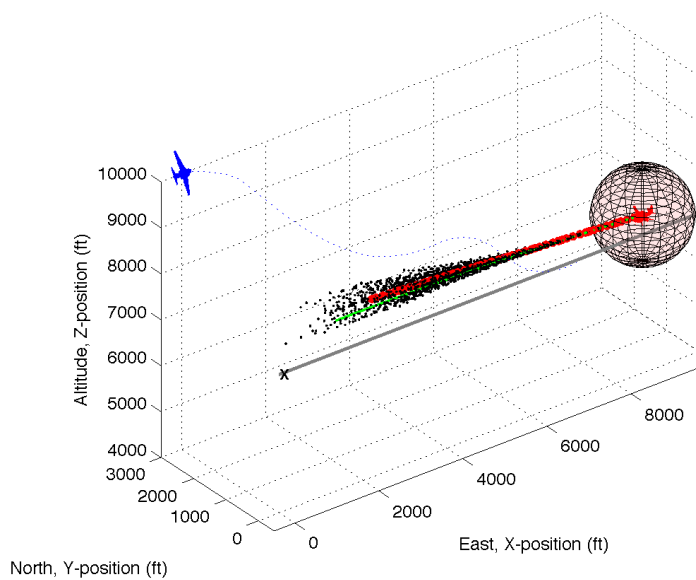
4.4 3D Results

This section provides the 3D results of the simulation scenario described in the previous section. In this scenario the optimization algorithm attempts to minimize the deviation from a 3D flightpath corridor using maximum control available (bounded in the constraint function only). This generates an optimal 3D path for the ownship to fly that satisfies the boundary, path, and dynamic constraints. For this simulation, the ownship starts the scenario at an altitude of 10,000 feet and flies at a constant velocity of 300 feet/second in an easterly direction (positive x-axis). The ownship attempts to intercept the 3D flightpath corridor located along the current flight path but laterally displaced by 3,000 feet to the south (y-axis) and located 3,000 feet below (z-axis) at an altitude of 7,000 feet. The intruder aircraft in this scenario starts at an altitude of 7,150 feet and flies in a westerly direction

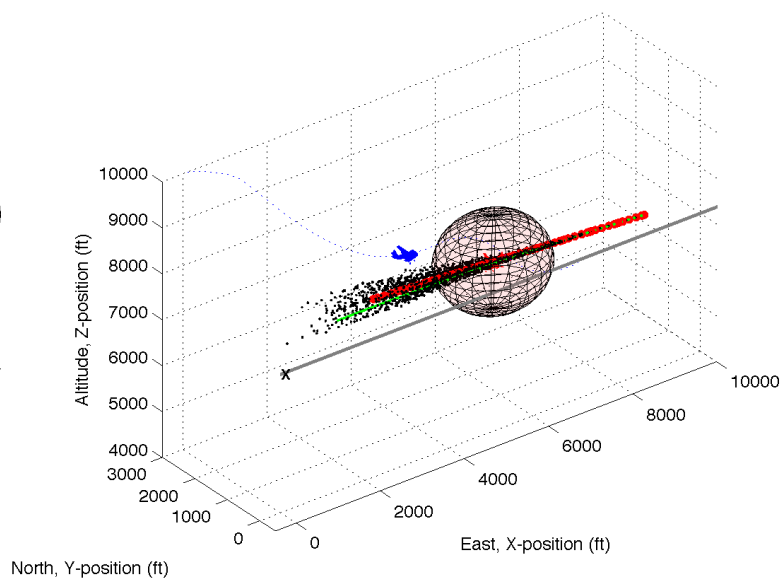
(negative x-axis) towards the ownship’s intended flight path. The intruder’s initial velocity is 200 feet/second in the west direction (x-axis), 20 feet/second in the north direction (y-axis), and zero feet/second in altitude (z-axis). Essentially, in this scenario the intruder flies towards the ownship to force a conflict. The initial slant range distance between the two aircraft is approximately 9,500 feet.

For the simulation results presented, the blue aircraft represents the ownship and the red aircraft represents the intruder aircraft. The blue circles represent the calculated optimal avoidance trajectory at each time interval and the red circles represent the particle filter’s 30 second estimate of the intruder’s future 3D trajectory. The light red-colored ellipsoid on the plot represents the particle filter derived uncertainty volume associated with the intruder’s position estimate at each time. The optimal control algorithm uses this uncertainty volume as an inequality path constraint to calculate an optimal avoidance trajectory. To assess system performance, the green line on the plot represents the “true” 3D intruder trajectory. This true trajectory is arbitrarily chosen for this example as one of the 1,000 random trajectories generated by the particle filter at time zero. The gray-colored line represents the ownship’s intended 3D flightpath corridor. The black dots on the plot represent the propagated particles estimate of the intruder’s position at each measurement update (1 Hz). To prevent cluttering the plot, the results display only 10% of the actual particles.

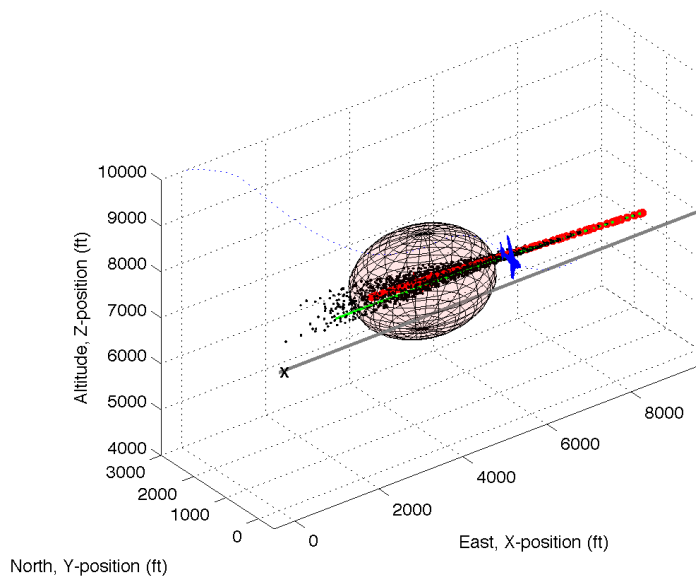
To establish a baseline for comparison and to demonstrate the feasibility of this approach, the results in Figure 4.1 on the following page graphically shows the algorithm’s performance without any measurement update during the entire 30 second time interval. In this scenario, the particle filter received an initial measurement (or observation) at time zero and used this information to calculate a 30 second estimate of the intruder’s position and uncertainty volume based on the standard deviations in the x, y, and z axis. As seen in Figure 4.1 on the next page, the algorithm appears to perform well in this scenario. The optimization algorithm successfully generated a collision avoidance trajectory that accounts for the particle filter’s estimate of the intruder uncertainty. Further, this optimal trajectory



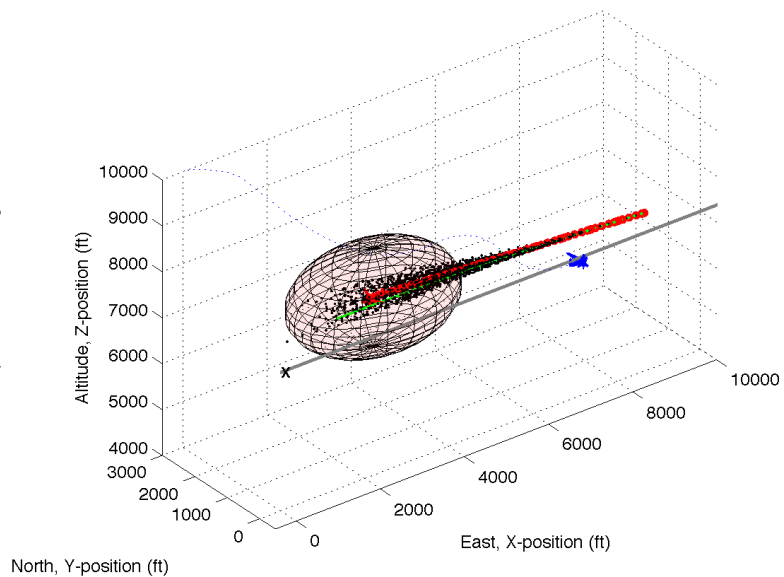
(a) Time 0 seconds



(b) Time 16.7 seconds



(c) Time 24.3 seconds



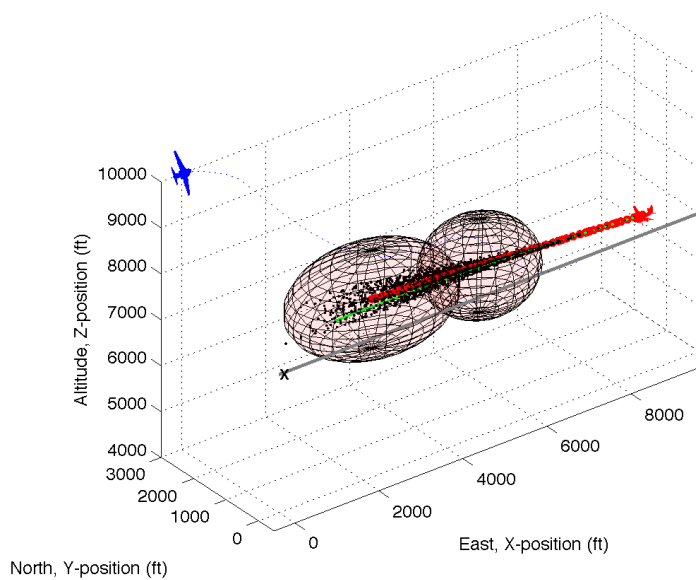
(d) Time 29.9 seconds

Figure 4.1: Time Series of Optimal Trajectory for Ownship (Blue) Avoiding the Intruder Aircraft's (Red) Uncertainty Volume While Minimizing Path Deviation with No Measurement Updates.

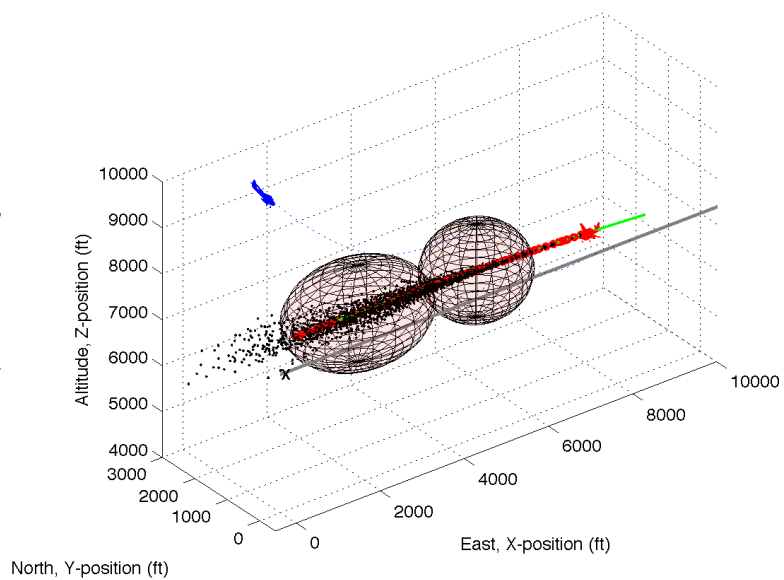
intuitively aligns with logic and successfully keeps the ownship away from the nominal or estimated intruder trajectory by the required separation distance of 1,000 feet.

At time zero, the ownship begins a right bank descending turn to intercept the desired 3D flightpath corridor (gray-line). While still in a descent as shown in panel (b) of Figure 4.1, the ownship approaches the intruder’s uncertainty volume from the northeast. The ownship then stops the descent and starts a slight climb while beginning a left bank turn in order to fly “behind” the intruder’s uncertainty volume. As the intruder’s uncertainty volume passes the ownship the ownship aircraft then turns back to the right and continues the descent in order to intercept the 3D flightpath corridor as shown in panel (c) of Figure 4.1. The minimum 3D separation distance between the ownship and the intruder’s nominal (estimated) trajectory is 1,042 feet and the minimum separation between ownship and the true intruder’s trajectory is 990 feet. The desired minimum separation distance for this scenario was 1,000 feet.

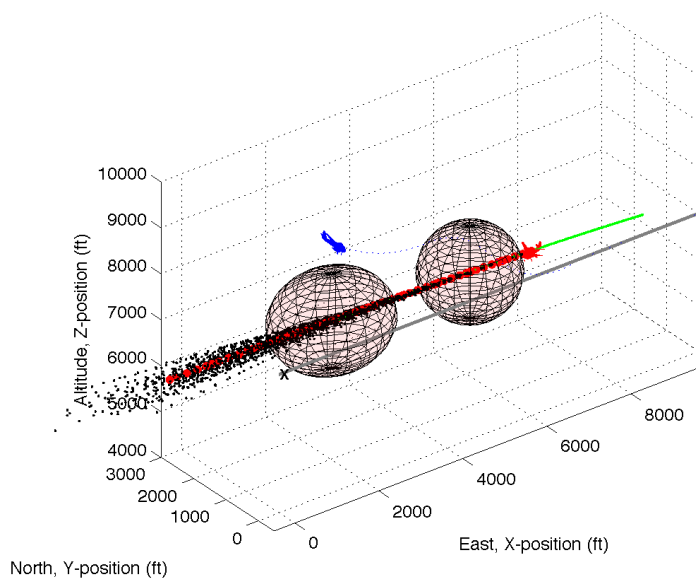
Despite not having any measurement updates during the 30 second time horizon, the algorithm performed well. The optimal trajectory satisfied the inequality path constraints of avoiding the intruder’s dynamically changing uncertainty volume while minimizing path deviations from the desired 3D flightpath corridor. However, in the absence of a measurement, the optimal trajectory was 10 feet less than desired from the true trajectory at the closest point. Nonetheless, to mitigate against extended periods such as this without a measurement update, the system designer can easily adapt this algorithm and adjust the rate of change for the uncertainty volume to increase the probability that the ownship remains outside of a defined region from the true intruder trajectory. Subsequently, as seen in Figure 4.1 on the preceding page, the particle filter’s initial mean estimates of the intruder’s 30 second flight path (red circles) does not necessarily match the intruder’s true 30 second trajectory (green line). Receiving regular measurement updates should improve the filter’s estimate of the intruder’s trajectory (that is, cause the red circles to more closely overlay the green line) and reduce the uncertainty volume surrounding the intruder.



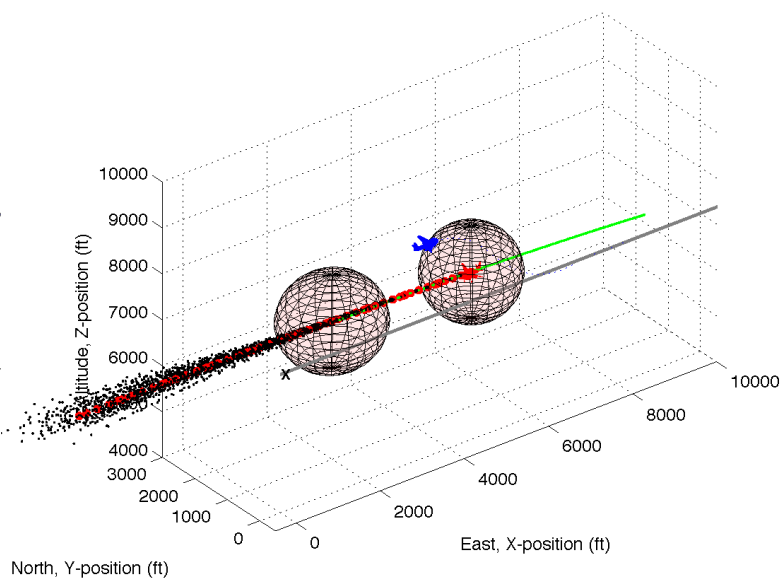
(a) Time 0 seconds



(b) Time 6 seconds



(c) Time 12 seconds



(d) Time 18 seconds

Figure 4.2: Time Series of Optimal Trajectory for Ownship (Blue) Avoiding the Intruder Aircraft's (Red) Uncertainty Volume While Minimizing Path Deviation, Showing the Effect of Measurement Updates at Time 18 and 30 Seconds for the Receding Horizon.

Incorporating measurement updates, the results in Figure 4.2 on the previous page visually depict the reduction in uncertainty and subsequent performance gain, that is, the ability for the ownship to get closer to the intruder's nominal trajectory while maintaining the required separation, and thus, minimize the deviation from the intended flightpath corridor. The scenario for these results are identical to the previous scenario for both the ownship and intruder aircraft. The only exception is that now the algorithm includes regular measurement updates at a 1 Hz rate using the measurement model described earlier in this section. Figure 4.2 on the preceding page helps visualize the performance benefits of measurement updates by first showing the particle filter's initial estimate of the intruder's uncertainty ellipsoid at the 18 and 30 second point on the trajectory based on the initial measurement update at time zero. The simulation then incorporates regular measurement updates at a 1 Hz rate. Following the measurement updates at 6, 12, and 18 seconds, separate time sequence panels in Figure 4.2 on the previous page display the intruder and ownship position immediately after the measurement update and depicts how the original uncertainty volumes associated with the 18 and 30 second ellipsoids shrink from the initial filter estimate generated at time zero. This comparison visually shows how the trajectory uncertainty ellipsoids contract with increased number of measurement updates.

Likewise, as discussed earlier, in panel (a) of Figure 4.2 on the preceding page, the initial particle filter's estimated trajectory (red circles) does not necessarily match the intruder's true 30 second trajectory (green line). However, in this scenario, as seen in panels (c) and (d) of Figure 4.2, after several measurement updates the particle filter estimate appears to more closely match the intruder's true 30 second trajectory. Table 4.2 lists the minimum 3D separation distance between the ownship and the intruder's nominal (estimated) trajectory and the minimum separation between ownship and the intruder's true trajectory for each measurement time in Figure 4.2. Again, the desired minimum separation distance is 1,000 feet.

Table 4.2: Minimum Intruder 3D Separation Distance

Time (seconds)	Nominal Intruder (feet)	True Intruder (feet)
0	1,042	990
6	1,017	1,037
12	1,015	1,003
18	1,001	1,001

The first column in Table 4.2 shows the amount of elapsed time from the start of the 30 second trajectory, and based on the 1 Hz update rate, this column also represents the total number of measurements since the start of the trajectory. Thus, in Table 4.2 the minimum separation distance for the true intruder’s trajectory starting at time zero with no measurement updates was 10 feet less than the desired minimum distance of 1,000 feet. However, this value represents only a single evaluation of a stochastic process, and therefore, is not conclusive. Yet, as previously discussed, system designers can characterize the estimation error and adapt this algorithm to increase the confidence that the true intruder’s trajectory remains within the estimated uncertainty volume. As seen in Table 4.2, for this particular simulation run the minimum separation distance from the true intruder’s trajectory at 6, 12, and 18 seconds were all greater than the desired minimum distance of 1,000 feet. Again, the separation distances at these particular times were only single realizations of a stochastic process; an accurate characterization of system performance will require Monte Carlo analysis. In general, as additional measurements became available the particle filter was able to replicate the expected nature of the intruder. Thus, with proper error characterization, system designers can set the appropriate tolerance to ensure the last column in Table 4.2 approaches the desired minimum separation distance.

The simulation results demonstrated that the direct orthogonal collocation formulation method using a particle filter and a fixed finite time horizon can be used to estimate an intruder’s position and dynamically calculate an uncertainty ellipsoid or keep-out region for the optimization algorithm to use as a path inequality constraint while minimizing deviation to a defined 3D path. In this limited simulation, the optimization algorithm and particle filter performed well. For instance, in Figure 4.2 on page 58, the particle envelope

(black dots) consistently surrounded the green line which demonstrates that the particles are modeling the true PDF of the intruder. Furthermore, the optimization algorithm successfully stayed outside of the defined keep-out region for the estimated intruder. An area for future research is to validate the implementation of the Singer model as applied in this research by comparing the random intruder trajectory used in this simulation against real aircraft performance data from the NAS environment. In addition, the Singer acceleration model assumes that the aircraft's x , y , and z positions operate independently, which is usually not the case for an aircraft. Thus, Blackman and Popoli [58] present other models such as the *Coordinated Turn Model* which accounts for the natural correlation between the aircraft's position states that occur during a coordinated turn maneuver, which may improve the estimation performance when modeling real-world aircraft data.

4.5 Conclusion

The collision avoidance scenarios in the results sections of the current and previous chapter showed a general framework for posing the collision avoidance problem for RPAs as an optimal control problem using stochastic inputs for the intruder. The next step is to extend the estimation problem to provide ‘uncertainty corridors’ that can be used as ‘no-fly’ regions for use as constraints.

V. Uncertainty Corridors for 3D Collision Avoidance

THIS CHAPTER presents a new method and algorithmic procedure for formulating uncertainty corridors in 3D for inclusion as time-varying inequality path constraints for the nonlinear optimal control problem. The proposed general approach is applicable to any underlying distribution, Gaussian or otherwise, and efficiently captures the intruder's predicted future locations as ellipsoids using a convex optimization problem based on Khachiyan's algorithm [79]. These ellipsoids are then smoothly interpolated to accurately identify a collision avoidance corridor for the airborne sense-and-avoid optimal control problem. This approach is successfully demonstrated on a representative SAA scenario.

This chapter⁴ contains eight sections. Section 5.1 describes the various dynamic, observation and estimation models used in this chapter and Section 5.2 demonstrates a simple 2D S-Turn scenario. Section 5.3 describes the algorithms used to efficiently model an intruder's posterior distribution to accurately identify a time-varying collision avoidance corridor for use as a path constraint in the nonlinear optimal control problem and Section 5.4 develops the necessary interpolation algorithm. Section 5.5 describes the optimal control problem formulation for an airborne collision avoidance application and then provides an overview of the stochastic 3D scenario used in this chapter. Sections 5.6 and 5.7 describe and then analyze the 3D simulation results. Finally, Section 6.14 highlights the planned follow-on research and summarizes the results for this chapter.

5.1 Model Development

This section describes the various dynamic, observation and estimation models used in this chapter. The intruder model consists of a 5-state coordinated turn model for the horizontal filter [58] and a decoupled 3-state Singer acceleration model [61] for the vertical filter. As noted in [58], this technique of using a second decoupled filter to model motion in the vertical z direction is common for a 3D aircraft application. The observation model

⁴Note: This chapter appears as a conference paper with the co-authors listed in [80]. This chapter is intended for archival journal submission.

consists of relative measurements between the intruder and ownship that are nonlinear combinations of the state estimates. The ownship model is a 5-state, 3 DOF nonlinear point mass model described in Section 2.2.3.2.

5.1.1 Intruder Model Development.

A fundamental necessity in solving the airborne collision avoidance problem is the need to estimate the current and future position of the intruder along with the need to accurately model how the probability regions associated with this position estimate change as a function of time. The previous chapter utilized a linear intruder model which did not take into account the natural correlation in x and y positions for a turning aircraft, and this model assumed a Gaussian distribution for the uncertainty probability regions associated with the predicted trajectory. An additional shortfall with this, and any linear model, is the inability to accurately estimate the future position of a turning aircraft. Therefore, in this current work we adapt and utilize a slightly modified version of the coordinated turn model as presented in [58]. The coordinated turn model takes into account the natural correlation of x and y positions in the 2D horizontal plane for a turning aircraft and has the added benefit of estimating an intruder's turn-rate (ω), and thus, the model has the ability to estimate the future position of a turning aircraft. Clearly, for a predictive collision avoidance application the ability to estimate the future position for a turning aircraft is necessary. Another advantage of estimating turn-rate with a particle filter implementation is that the system designer for an airborne collision avoidance application may potentially be able to utilize a single model instead of relying on an interactive multiple model (IMM) approach [81] that utilizes a turn model ($\omega \neq 0$) and a linear model ($\omega = 0$) to estimate the dynamic performance of aircraft in the NAS.

5.1.1.1 Coordinated Turn Model for Horizontal Plane.

The following derivation of the coordinated turn model is adapted from [58, 82]. In the horizontal plane, the five states for the coordinated turn model are x , y , v_x , v_y , and ω . The state equations for the coordinated turn model appears below. This derivation assumes

a constant turn rate (ω) and constant speed (s).

$$x(k+1) = x(k) + \int_{t_k}^{t_{k+1}} v_x(\tau|t_k) d\tau \quad (5.1)$$

$$= x(k) + s \int_0^T \cos(\chi + \omega\tau) d\tau \quad (5.2)$$

$$= x(k) + sT [SW \cos \chi - CW \sin \chi] \quad (5.3)$$

$$= x(k) + T [SW v_x(k) - CW v_y(k)] \quad (5.4)$$

$$y(k+1) = y(k) + sT [SW \sin \chi + CW \cos \chi] \quad (5.5)$$

$$= y(k) + T [CW v_x(k) + SW v_y(k)] \quad (5.6)$$

where, the terms SW and CW define the coefficients in equations (5.3) - (5.6)

$$SW \triangleq \frac{\sin(\omega T)}{\omega T}, \quad CW \triangleq \frac{1 - \cos(\omega T)}{\omega T} \quad (5.7)$$

and the horizontal velocity values are defined as

$$v_x = s \cos \chi, \quad v_y = s \sin \chi \quad (5.8)$$

where χ is the intruder's heading angle and $\omega = \dot{\chi}$ = intruder's turn rate. Likewise, the velocity state equations appear as

$$v_x(k+1) = s \cos(\chi + \omega T) = v_x(k) \cos \omega T - v_y(k) \sin \omega T \quad (5.9)$$

$$v_y(k+1) = s \sin(\chi + \omega T) = v_x(k) \sin \omega T + v_y(k) \cos \omega T \quad (5.10)$$

Equations (5.4) - (5.10) represent the dynamic model for an aircraft flying a circular turn in the horizontal plane [58].

For the coordinated turn model in the work herein, the turn-rate state is modeled as a first-order Gauss-Markov (FOGM) random process rather than a constant. As such, this state changes due to both an additive white Gaussian noise and a maneuver time constant parameter. This maneuver time constant is analogous to the maneuver time constant development in the Singer acceleration model [61]. The idea is that since the collision avoidance algorithm in this application requires predicting an intruder's trajectory

for up to 30 seconds into the future, the maneuver time constant allows the estimated turn rate to appropriately transition back to a zero-mean value over time to account for greater uncertainty during extended trajectory propagation periods. The derivation of the model in [58] assumes a random walk for the propagation of ω ; however, in our problem formulation we provide the optimal control problem a 30-second predicted trajectory, which is equivalent to the filter producing a position estimate for 30 seconds without a measurement update. Subsequently, a random walk implementation does not necessarily represent the true future trajectory of the aircraft. Therefore, the following equations show the derivation for this first-order Gauss-Markov process used to modify the coordinated turn model, which also appears in [83].

In continuous time, the state equation for the FOGM process turn-rate state (ω) appears as

$$\dot{\omega} = -(1/\tau)\omega(t) + w(t) \quad (5.11)$$

where τ is the maneuver time constant associated with the turn rate state and w is zero mean white additive Gaussian noise with

$$E[w(t)w(t+\tau)] = Q\delta(\tau) \quad \text{where} \quad Q = \frac{2\sigma_\omega^2}{\tau} \quad (5.12)$$

and σ_ω^2 is the variance associated with the turn rate state (ω). The discrete time state transition matrix for ω appears as

$$\Phi_{\text{FOGM}}(\Delta t) = e^{-\Delta t/\tau} \quad (5.13)$$

Likewise, the discrete-time noise Q_{k_ω} appears as:

$$\begin{aligned} Q_{k_\omega} &= \sigma_\omega^2 \int_{t_{i-1}}^{t_i} \Phi^2(t_i, \tau) d\tau \\ &= \sigma_\omega^2 \int_{t_{i-1}}^{t_i} e^{-\frac{2}{\tau}(t_i-\tau)} d\tau \\ &= \sigma_\omega^2 \left(1 - e^{-\frac{2\Delta t}{\tau}}\right) \end{aligned} \quad (5.14)$$

Therefore, the discrete-time five-state coordinated turn model appears as [82]:

$$\mathbf{x}_{k+1} = \mathbf{F}_{x_k} \mathbf{x}_k + \mathbf{w}_k = \begin{bmatrix} 1 & 0 & \frac{\sin(\omega_k \Delta t)}{\omega_k} & \frac{\cos(\omega_k \Delta t) - 1}{\omega_k} & 0 \\ 0 & 1 & \frac{1 - \cos(\omega_k \Delta t)}{\omega_k} & \frac{\sin(\omega_k \Delta t)}{\omega_k} & 0 \\ 0 & 0 & \cos(\omega_k \Delta t) & -\sin(\omega_k \Delta t) & 0 \\ 0 & 0 & \sin(\omega_k \Delta t) & \cos(\omega_k \Delta t) & 0 \\ 0 & 0 & 0 & 0 & \Phi_{\text{FOGM}} \end{bmatrix} \mathbf{x}_k + \mathbf{w}_k \quad (5.15)$$

In equation (5.15), \mathbf{w}_k is the additive discrete time noise process with a noise strength of \mathbf{Q}_k . With the exception of the derivation of $Q_{k\omega}$ shown in equation (5.14), the derivation of \mathbf{Q}_k , appears in [81] as:

$$\mathbf{Q}_k = q_k \begin{bmatrix} \frac{\Delta t^4}{4} & 0 & \frac{\Delta t^3}{2} & 0 & 0 \\ 0 & \frac{\Delta t^4}{4} & 0 & \frac{\Delta t^3}{2} & 0 \\ \frac{\Delta t^3}{2} & 0 & \Delta t^2 & 0 & 0 \\ 0 & \frac{\Delta t^3}{2} & 0 & \Delta t^2 & 0 \\ 0 & 0 & 0 & 0 & \frac{Q_{k\omega}}{q_k} \end{bmatrix} \quad (5.16)$$

For aircraft flying in the NAS, a standard rate turn is 3 deg/sec; however, due to performance limitations some larger aircraft use half-standard rate turns of 1.5 deg/sec. In this chapter we selected the value of σ_ω^2 equal to be 1 deg²/sec² based on the work by [84] who analyzed the turn rate of 193 million recorded radar samples for aircraft flying in the NAS. This data included aircraft flying straight and aircraft turning. Based on this recorded radar data, a turn-rate of 6 deg/sec approximates six-sigma of this data. Finally, the value q_k in equation (5.16) is used as a tuning parameter and represents the random acceleration variance.

5.1.1.2 Singer Acceleration Model for Vertical Motion.

The derivation of the Singer acceleration model appears in [61]. In the vertical direction, the three states for this model are position (z), velocity (v_z) and acceleration (a_z). This model assumes the intruder's acceleration $a(t)$ to be a zero-mean first-order stationary Markov process with autocorrelation $R_a(\tau) = E[a(t)a(t + \tau)] = \sigma_z^2 e^{-\alpha|\tau|}$, or equivalently, power spectrum $S(\omega) = 2\alpha\sigma_z^2/(\omega^2 + \alpha^2)$ [59].

For the Singer acceleration model, the discrete-time propagation model appears as [61]

$$\mathbf{z}_{k+1} = \mathbf{F}_{z_k} \mathbf{z}_k + \mathbf{w}_k = \begin{bmatrix} 1 & \Delta t & \frac{(\alpha \Delta t - 1 + e^{-\alpha \Delta t})}{\alpha^2} \\ 0 & 1 & \frac{(1 - e^{-\alpha \Delta t})}{\alpha} \\ 0 & 0 & e^{-\alpha \Delta t} \end{bmatrix} \mathbf{z}_k + \mathbf{w}_k \quad (5.17)$$

In this model, Δt represents the sample rate and the parameter $\alpha = 1/\tau_m$ is the reciprocal of the maneuver time constant τ_m . For an aircraft application, a τ_m of approximately 60 seconds represents a “lazy turn” and a τ_m between 10 – 20 seconds represents a highly maneuvering target [59]. In this chapter, we selected τ_m equal to 60 seconds. The discrete-time noise strength matrix, \mathbf{Q}_k , appears as [61]

$$\mathbf{Q}_k = 2\alpha\sigma_m^2 \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{12} & q_{22} & q_{23} \\ q_{13} & q_{23} & q_{33} \end{bmatrix} \quad (5.18)$$

where

$$\begin{aligned} q_{11} &= \frac{1}{2\alpha^5} \left[1 - e^{(-2\alpha\Delta t)} + 2\alpha\Delta t + \frac{2\alpha^3\Delta t^3}{3} - 2\alpha^2\Delta t^2 - 4\alpha\Delta te^{(-\alpha\Delta t)} \right] \\ q_{12} &= \frac{1}{2\alpha^4} \left[e^{(-2\alpha\Delta t)} + 1 - 2e^{(-\alpha\Delta t)} + 2\alpha\Delta te^{(-\alpha\Delta t)} - 2\alpha\Delta t + \alpha^2\Delta t^2 \right] \\ q_{13} &= \frac{1}{2\alpha^3} \left[1 - e^{(-2\alpha\Delta t)} - 2\alpha\Delta te^{(-\alpha\Delta t)} \right] \\ q_{22} &= \frac{1}{2\alpha^3} \left[4e^{(-\alpha\Delta t)} - 3 - e^{(-2\alpha\Delta t)} + 2\alpha\Delta t \right] \\ q_{23} &= \frac{1}{2\alpha^2} \left[e^{(-2\alpha\Delta t)} + 1 - 2e^{(-\alpha\Delta t)} \right] \\ q_{33} &= \frac{1}{2\alpha} \left[1 - e^{(-2\alpha\Delta t)} \right] \end{aligned} \quad (5.19)$$

In this chapter we selected the value of σ_z^2 based on the intruder’s estimated vertical velocity.

5.1.2 Observation Model.

The sensor measurements, \mathbf{z}_k , in this chapter are relative measurements between the ownship and the intruder that occur at time k . In this simulation the sensor is a radar system located onboard the ownship aircraft. At each measurement time k , the algorithm performs relative slant range (z_R), radial range rate (z_{v_R}), azimuth (z_{az}), and elevation

angle (z_{el}) measurements of the intruder with respect to the ownship. The measurement equations appear as [2]

$$\mathbf{h}(\mathbf{x}) = \begin{bmatrix} z_R \\ z_{vR} \\ z_{az} \\ z_{el} \end{bmatrix} = \begin{bmatrix} \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \\ \frac{\Delta x \Delta v_x + \Delta y \Delta v_y + \Delta z \Delta v_z}{\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}} \\ \tan^{-1} \frac{\Delta y}{\Delta x} \\ \sin^{-1} \frac{\Delta z}{\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}} \end{bmatrix} \quad (5.20)$$

where Δ represents the relative difference in position or velocity between the ownship and the intruder. These measurements of the intruder are nonlinear functions of the intruder's state, $\mathbf{h}(\mathbf{x}_k)$, and subject to noise, thus are modeled according to

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k) + \boldsymbol{\nu}_k \quad (5.21)$$

where $\boldsymbol{\nu}$ is zero-mean white Gaussian noise with noise strength,

$$\mathbf{E}[\boldsymbol{\nu}_j^T \boldsymbol{\nu}_k] = \mathbf{R}_m \delta_{jk} \quad (5.22)$$

where δ_{jk} is the Kronecker delta function. The noise autocorrelation, \mathbf{R}_m , indicates the uncertainty associated with the sensor. The noise strength associated with each measurement appears along the diagonal elements of the matrix \mathbf{R}_m . The simulation described in this chapter uses the following noise autocorrelation matrix, which are representative of radar performance as described in [9].

$$\mathbf{R}_m = \begin{bmatrix} (30 \text{ ft})^2 & 0 & 0 & 0 \\ 0 & (10 \text{ ft/sec})^2 & 0 & 0 \\ 0 & 0 & (0.5 \text{ deg})^2 & 0 \\ 0 & 0 & 0 & (0.5 \text{ deg})^2 \end{bmatrix} \quad (5.23)$$

5.2 2D Coordinated S-Turn Scenario

This section demonstrates the performance of the particle filter as described in Section 4.2 using the modified coordinated turn model shown in equations (5.15) and (5.16)

along with the observation model in equations (5.20) and (5.21) to produce estimates of an intruder’s position. The purpose of this 2D scenario is twofold: First, demonstrate that the particle filter using the modified coordinated turn model can estimate the performance of a turning and straight flying aircraft in the NAS. Second, demonstrate the distribution of the point cloud following extended periods without a measurement update in order to gain insight into how best to model these probability regions as inequality path constraints or collision avoidance corridors for the optimal control problem. The 2D scenario had a fixed altitude and did not include the Singer acceleration model for the vertical channel. We include the vertical channel later in Section 5.5 for the optimal control problem formulation.

In this scenario, the intruder flies at a constant speed of 250 ft/sec and travels from east to west (or right to left on the page). The ownship flies straight at a constant speed from west to east towards the intruder while the intruder performs an S-Turn pattern about the ownship’s projected flight path alternating between 30-second straight legs and 30-second turning legs. The turn legs use a standard rate turn of 3 deg/sec and alternate turn direction from right to left. The length of the straight legs are 7,500 feet and the radius for the turning legs is approximately 6,600 feet. Although aircraft do not typically fly S-Turn patterns in the NAS, this scenario is intentionally setup to challenge the robustness of the model.

In this scenario, we set the maneuver time constant described by equation (5.11) equal to 30 seconds. Although the filter propagates 50,000 particles every second, in this scenario the filter only receives a measurement update once every 10 seconds. This extended period without a measurement purposely mirrors the constraint formulation for the optimal control problem where the “keep-out corridor” or “no-fly zone” which is the inequality path constraint based on the intruder’s position estimate propagated for 30 seconds without a measurement update.

In 2D, Figure 5.1 is a time-sequenced quad chart showing the position estimates at 70, 140, and 220 seconds. For clarity, the ownship is intentionally not displayed in this figure. The dark blue trajectory lines in this figure indicate the truth trajectory for the straight legs

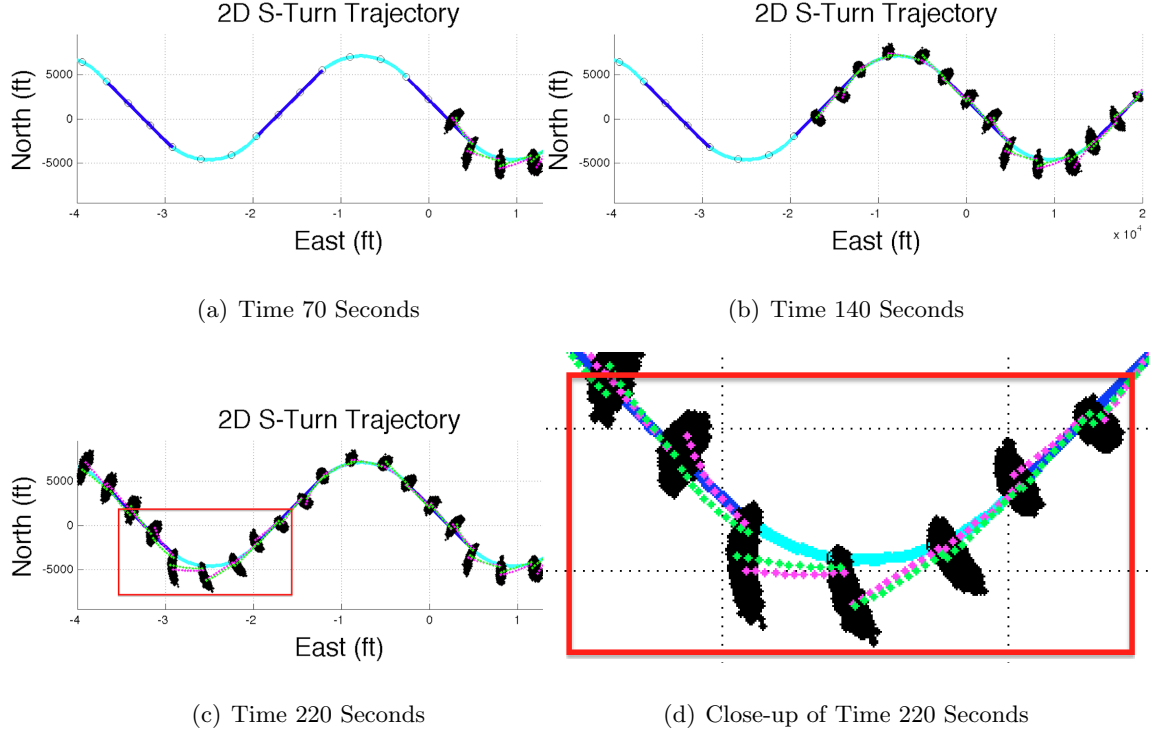


Figure 5.1: 2D S-Turn Trajectory

and the light-blue or cyan color indicate the truth trajectory for the turning legs. The black point clouds are the propagated position estimates from the particle filter that define the intruder's probability regions. Although the filter propagates 50,000 particles every second, to avoid cluttering the plot we only display the point cloud estimates from the filter every 10 seconds (just prior to the measurement update). Nevertheless, to facilitate insight into the filter's performance between measurement updates, we display the position estimate from two arbitrarily selected particles, one in green and one in magenta, every second. To better help see these details, in Panel (d) of Figure 5.1 we zoom in and show a close-up of the red-highlighted area in panel (c).

The two main takeaways from this demonstration are: (1) the shape of the point cloud estimates depend on geometry and model dynamics and are not necessarily Gaussian, and (2) the modified coordinated turn model successfully follows the S-Turn trajectory, that is, the point cloud estimate throughout the simulation intersects with the blue- and

cyan-colored truth trajectories. Thus, if we can efficiently model these probability regions produced by the filter we can use this model as the inequality path constraints for the optimal control problem to effectively identify collision avoidance corridors.

5.3 Collision Avoidance Constraint Modeling

5.3.1 *Algorithm Considerations and Development.*

The first step needed in order to use the point cloud distributions to set collision avoidance corridors for the optimal control problem is to identify an algorithm that can efficiently capture and model these time-varying distributions. The desired attributes are that this algorithm be mathematically efficient, tractable, and able to quickly compute at any instant in time if a given trajectory falls inside or outside of the corridor. With these considerations in mind, when viewing the point cloud distributions in Figure 5.1 one can conclude that a minimum volume enclosing ellipse algorithm would make an ideal choice to efficiently capture the time-varying point cloud distribution at each time step. Although a sphere is another shape that could also capture the point cloud distribution, a sphere is far too conservative since in most applications, like the SAA problem, the probability region tends to have different distributions in each direction. Nevertheless, in the rare case where the distributions along each axes are equal, the resulting sphere is just a special case of an ellipsoid where the radii of the principal axes are all equal.

Since the direct orthogonal collocation method that solves the optimal control problem discretizes the trajectory at non-equal time intervals that are independent of the measurement updates, in addition to merely capturing this probability region we also require a method to smoothly interpolate from one region to the next. From the results in Figure 5.1, a reasonable assumption in determining this interpolation method is that the point cloud distributions are continuous and transition smoothly between measurement updates. As a result, we selected a ‘spherical linear interpolation’ or Slerp as first introduced by Shoemake [85]. The Slerp algorithm performs the interpolation by referencing a unit sphere and uses the shortest arc length as the path for the interpolation; consequently, this path results in a constant angular velocity [85]. Based on this interpolation scheme,

for any arbitrary time between t_i to t_f the optimal control problem can now determine an intermediate ellipsoid that accurately characterizes the probability region associated with the intruder's position at that time. In formulating the inequality constraints for the optimal control problem, the algorithm takes advantage of the eigenstructure of the minimum volume enclosing ellipsoid to define the keep-out corridor and uses the quaternions of the ellipsoids to perform the interpolation. We refer to the combination of both the interpolation and enclosing ellipsoid algorithms as SLIMVEE for spherical linear interpolation of minimum volume enclosing ellipsoid. The following section provides an overview of the SLIMVEE algorithm as it relates to the airborne SAA nonlinear optimal control problem.

5.3.2 Algorithm Overview.

This section provides an overview of the use of eigenstructures and quaternions to interpolate an ellipsoid, as a function of time, that bounds a 3D probability region associated with an intruder's trajectory from some time initial (t_i) to some time final (t_f) for use as an inequality path constraint in a nonlinear optimal control problem. In general, the SLIMVEE algorithm consists of four steps. The flowchart in Figure 5.2 graphically details this process.

The first step is to capture the probability region at (t_i) and (t_f) using an optimization algorithm. This optimization algorithm computes the minimum volume enclosing ellipsoid (MVEE) for a 3D probability region associated with the intruder's position represented as point cloud distributions generated from a sequential Monte Carlo process based on the dynamics model and measurement updates. This minimum volume enclosing ellipsoid algorithm is based on Khachiyan's algorithm as described by [79]. For computational efficiency, as part of this step we first employ a convex hull algorithm that eliminates repeated and interior points leaving only unique boundary points defining the point cloud distribution.

We next use a singular value decomposition (SVD) to decompose each MVEE into singular vectors and singular values. The vectors define the orthogonal rotation matrix that maps each ellipsoid's local reference frame to the inertial or global reference frame.

The singular values associated with each matrix identifies the radius of the three principal axes of each ellipsoid. The third step is to represent the orientation of the initial and final ellipsoids as quaternions and evaluate the inner product in order to determine the orientation that results in the minimum rotation angle when interpolating from the initial to final ellipsoid. (This step eliminates the 180° ambiguity on the orientation of the ellipsoid about each axis.)

The final step is to interpolate the distribution using a Slerp algorithm. Based on this interpolation, for any arbitrary time between t_i to t_f the optimal control problem can now determine an intermediate ellipsoid that accurately characterizes the probability region associated with the intruder's position at that time. We then store the results of these interpolated ellipsoids as new matrices in quadratic form (shown in Section 5.3.4) for use as inequality path constraints in the nonlinear optimal control problem. The flowchart in Figure 5.2 graphically details this process.

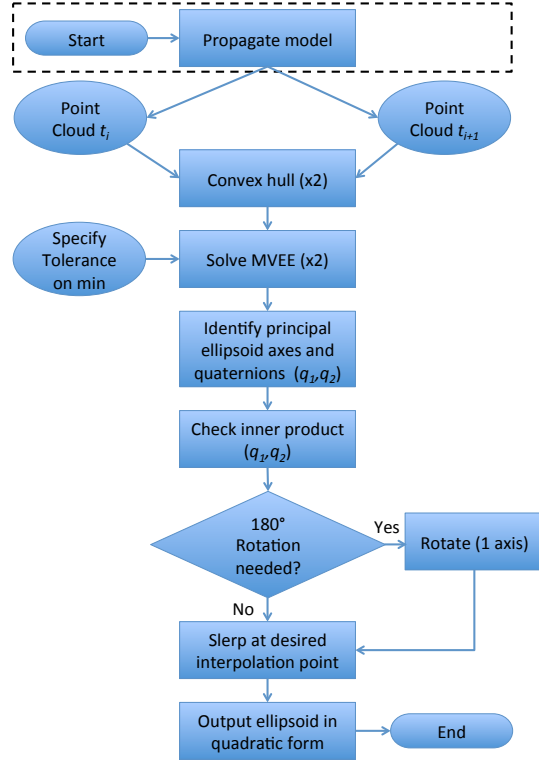


Figure 5.2: SLIMVEE Flowchart

5.3.3 Generating the Distribution.

The following example shows how to generate and capture 3D point cloud distributions based on the combined 8-state coordinated turn and the Singer acceleration models, equations (5.15) - (5.17), for an aircraft in a level, standard-rate (3 deg/sec) left-turn. Similar to the results in Section 5.1-5.2, applying a 1 Hz propagation rate and 10,000 particles per second to the combined 8-state model produces 3D point clouds representing the distribution of the intruder's position as a function of time. Replicating the inequality

constraint formulation for the optimal control problem, the filter generates a 30-second predicted trajectory by propagating the intruder's position based on the estimated states, model dynamics, and process noise. In this example, the filter initializes with the intruder performing a standard rate left turn, $\omega = 3$ deg/sec. The filter then propagates the intruder's position for the next 30 seconds using a maneuver time constant of 30 seconds. In Figure 5.3, the distribution for the intruder's position appears as red dots at 8 seconds (t_i) and as black dots at 11 seconds (t_f) into the 30-second propagation.

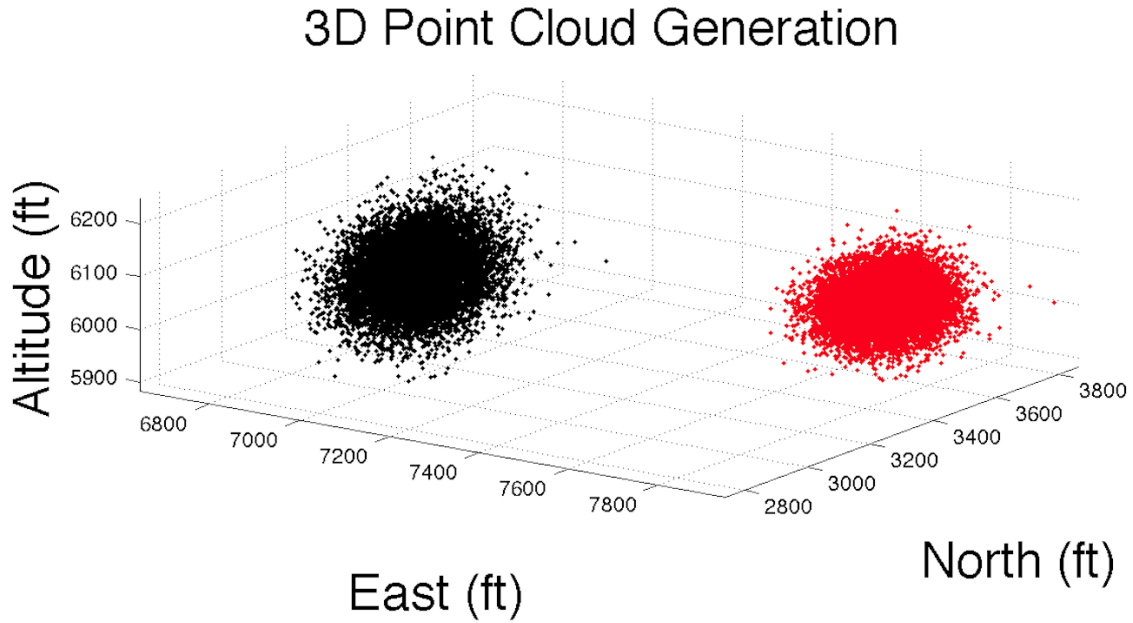


Figure 5.3: Point Cloud at 8 & 11 Seconds

5.3.4 Capture Probability via Minimum Volume Enclosing Ellipsoid.

The previous section described the process for generating the 3D point cloud distributions associated with a maneuvering aircraft. This section describes the process for capturing these distributions at (t_i) and (t_f) using a minimum volume enclosing ellipsoid (MVEE) optimization algorithm as described in [79, 86]. To reduce computation time prior to computing the MVEE, we first employ a convex hull algorithm [87] that identifies the minimum unique convex points that define the point cloud boundaries. Using the results of

this convex hull algorithm, we then apply the MVEE optimization algorithm. By definition, an ellipsoid in quadratic form appears as [86],

$$\mathcal{E} = \{\mathbf{x} \in \mathbb{R}^{n \times 1} \mid (\mathbf{x} - \mathbf{c})^T \mathbf{A} (\mathbf{x} - \mathbf{c}) \leq 1\} \quad (5.24)$$

where $\mathbf{c} \in \mathbb{R}^{n \times 1}$ is the center of the ellipsoid \mathcal{E} and $\mathbf{A} \in \mathbb{R}^{n \times n}$ and is a real symmetric matrix. Through the use of SVD, it is well-know that this matrix \mathbf{A} can be represented by a rotation matrix (\mathbf{R}) and diagonal matrix of eigenvalues ($\boldsymbol{\sigma}$), where the eigenvalues contain the magnitude of the ellipsoid's principal axes such that,

$$\mathbf{A} = \mathbf{R} [\boldsymbol{\sigma}] \mathbf{R}^T \quad (5.25)$$

The volume of \mathcal{E} is defined as [86]

$$\text{Vol}(\mathcal{E}) = \frac{v_0}{\sqrt{\det(\mathbf{A})}} = v_0 \det(\mathbf{A}^{-1})^{\frac{1}{2}} \quad (5.26)$$

“where v_0 is the volume of the unit hypersphere in dimension n ” [86]. As a result, a “natural formulation” of the minimum volume enclosing ellipsoid (MVEE) appears as [86],

$$\text{minimize: } \det(\mathbf{A}^{-1}) \quad (5.27)$$

$$\text{subject to: } (\mathbf{x}_i - \mathbf{c})^T \mathbf{A} (\mathbf{x}_i - \mathbf{c}) \leq 1 \quad \text{where } i = 1, 2, 3 \quad (5.28)$$

$$\mathbf{A} > 0 \quad (5.29)$$

However, this formulation is not a convex optimization problem [79, 86]. By applying a change of variables to equations (5.27) - (5.29) we can change this optimization problem into a convex optimization problem and solve this problem for the MVEE. Details of the derivation and formulation of this algorithm are in [79, 86].

Given two point cloud distributions at (t_i) and (t_f) this optimization algorithm returns two new matrices, \mathbf{A}_i at (t_i) and \mathbf{A}_f at (t_f) , and two vectors, \mathbf{c}_i and \mathbf{c}_f , which represent the MVEE in quadratic form along with the centroid for each ellipsoid at time initial and time final, respectively. Figure 5.4 on the following page shows the results of applying this MVEE optimization algorithm on the point cloud distributions in Figure 5.3. The blue, green, and red lines in Figure 5.4 originate at the centroid of the ellipsoid identifying the principal axes

as well as highlighting the orientation of each ellipsoid. Figure 5.4 shows that the MVEE optimization algorithm can clearly define specific 3D keep-out regions for the optimal control problem at discrete values of t ; however, we need a 3D keep-out corridor defined for all possible values of time. This requires a smooth ellipsoid interpolation algorithm.

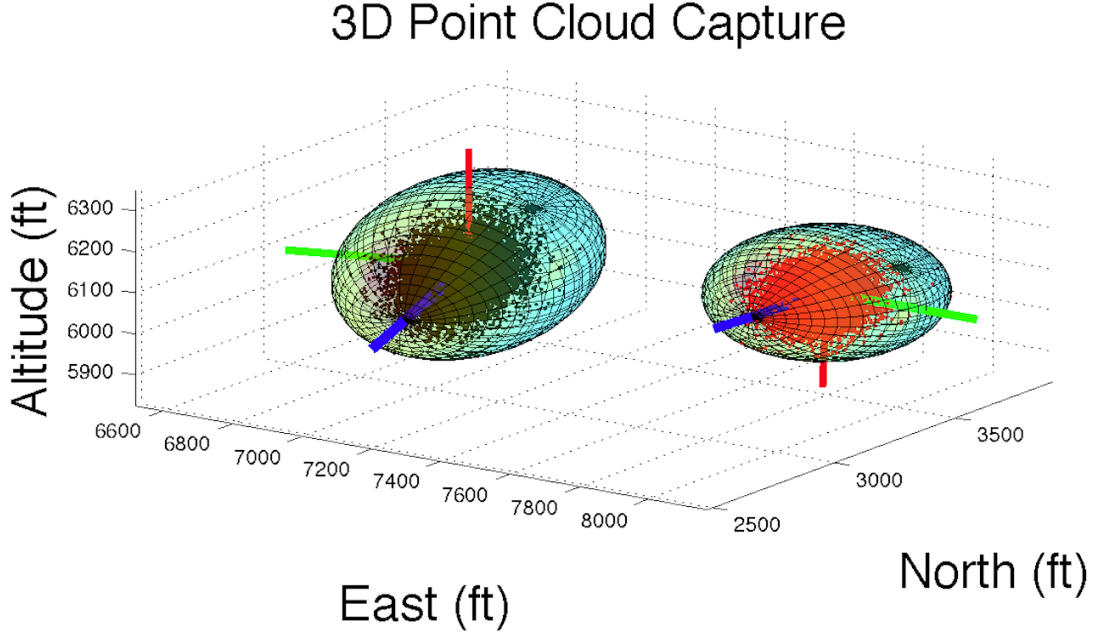


Figure 5.4: Point Cloud Capture at 8 & 11 Seconds

5.3.5 Ensuring Correct Ellipsoid Orientation.

A nuance which needs to be addressed is the orientation of the final MVEE when interpolating between two ellipsoids. Based on symmetry, a 180-deg rotation about any principal axis will not change the apparent position or volume of an ellipsoid. For example, in Figure 5.4 the MVEE at t_i (enclosing red dots) is orientated such that the longest principal axis (in blue) is pointing left and towards the reader, the middle principal axis (in green) is pointing right and towards the reader, and the smallest principal axis (in red) is pointing down. However, the MVEE at t_f (enclosing black dots) is orientated such that the longest principal axis (in blue) is still pointing left and towards the reader but the middle principal axis (in green) is pointing left and away from the reader, while the smallest principal axis

(in red) is pointing up. Because a 180-deg rotation about any principal axis does not alter the position or volume of an ellipsoid, the algorithm in this chapter evaluates the inner product of the quaternions to determine the proper orientation of the final ellipsoid that minimizes the quaternion error, which in turn finds the minimum angle between ellipsoids for use when interpolating from t_i to t_f . Figure 5.5 shows the results of applying this inner product evaluation to the final ellipsoid in Figure 5.4. Figure 5.6 on the following page shows the results of applying the MVEE optimization algorithm with the inner product evaluation to the entire 30-second trajectory described in Section 5.3.3. Again, this 30-second trajectory is based on a 3 deg/sec left turn with a maneuver time constant of 30 seconds as described in equation (5.12). To account for additional uncertainty such as the physical dimensions of an aircraft, we set the minimum radius to 500 feet for all ellipsoid axes. The results in Figure 5.6 clearly defines a 3D keep-out corridor for the optimal control problem; however, since the corridor is only defined for discrete values of t , we now must interpolate to define the keep-out corridor for all possible values of time.

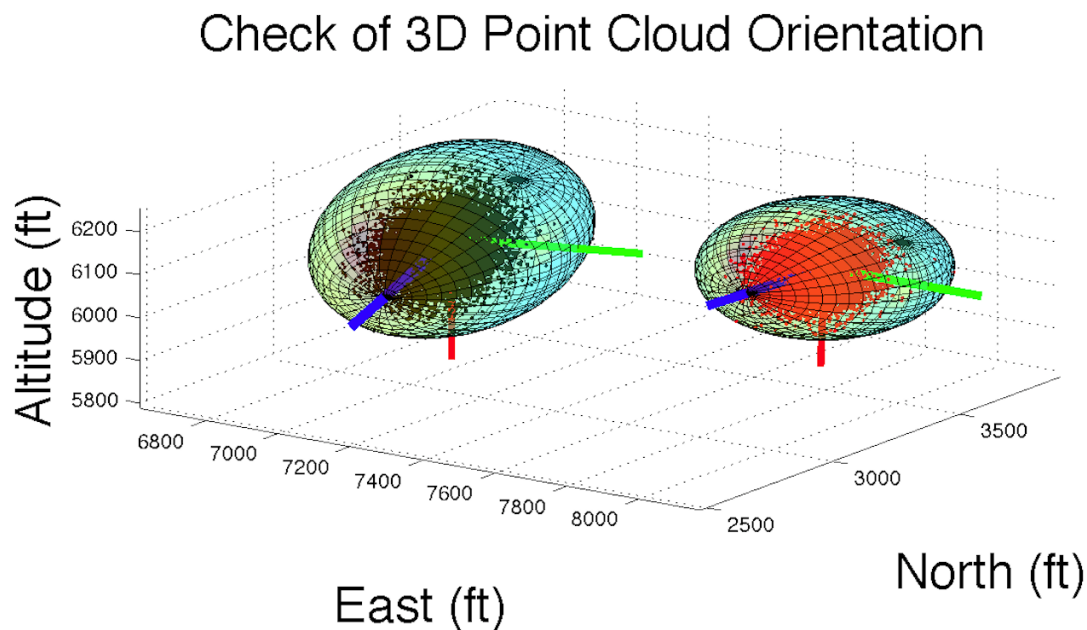


Figure 5.5: Determining Correct Ellipsoid Orientation for Interpolation

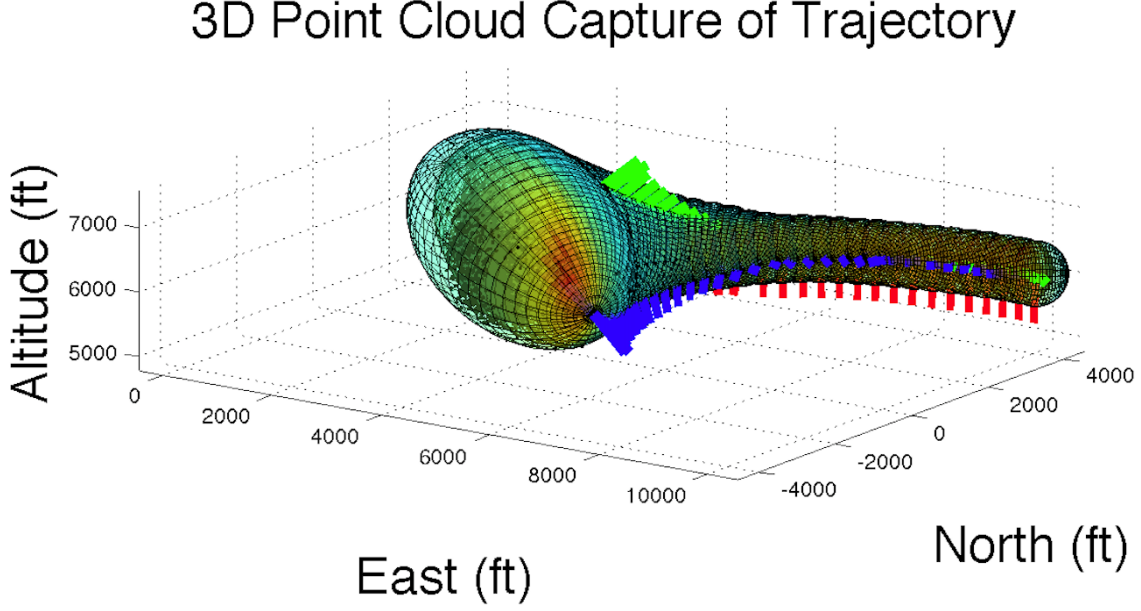


Figure 5.6: MVEE's for 30-Second Left Turning Intruder Trajectory

5.4 Interpolation Algorithm

After defining the MVEE for two point cloud distributions we can now apply an interpolation algorithm to define the probability region between these distributions. In order to accomplish this interpolation we first use the rotation matrix to represent each ellipsoid's orientation via quaternions. The quaternion, \mathbf{q} , is an orthonormal vector in \mathbb{R}^4 that consists of a scalar portion, q_0 , and a vector portion such that [88, 89]

$$\mathbf{q} = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k} = \begin{bmatrix} q_0 \\ \vec{q} \end{bmatrix} \quad (5.30)$$

$$q_0 = \cos\left(\frac{\theta}{2}\right) \quad (5.31)$$

$$\vec{q} = \vec{e} \cdot \sin\left(\frac{\theta}{2}\right) \quad (5.32)$$

where \vec{e} is the unit eigenaxis vector and θ is the rotation angle about this vector [89]. To interpolate between \mathbf{q}_1 , the quaternion at t_i , to \mathbf{q}_2 , the quaternion at t_f we use the Slerp algorithm. This algorithm performs the interpolation by referencing a unit sphere and takes

the shortest arc length as the path for the interpolation such that [85]

$$\text{Slerp}(\mathbf{q}_1, \mathbf{q}_2; u) = \frac{\sin(1-u)\theta}{\sin(\theta)}\mathbf{q}_1 + \frac{\sin u\theta}{\sin(\theta)}\mathbf{q}_2 \quad (5.33)$$

where u is the interpolation time and $\mathbf{q}_1 \cdot \mathbf{q}_2 = \cos \theta$. Thus, the algorithm to interpolate between two ellipsoids in order to identify the probability region for the optimal control problem requires three inputs: (1) the matrices \mathbf{A}_i and \mathbf{A}_f , which represents the MVEE in quadratic form at time initial and time final, respectively (2) the centroid (\mathbf{c}) for each ellipsoid at these times, and (3) the interpolation time, u . The first step of this algorithm is to decompose the matrices \mathbf{A}_i and \mathbf{A}_f into their respective rotation matrices (\mathbf{R}_k) and diagonal matrices of eigenvalues ($\boldsymbol{\sigma}$). A check on the second (or ‘final’) rotation matrix by successively rotating each axis by 180 degrees is performed to minimize the quaternion error to ensure a minimum rotation angle. Next, the quaternions from the rotation matrices (\mathbf{R}_k) and the lengths of the ellipsoids’ radii from the diagonal matrices of eigenvalues are used as the initial and final conditions for the Slerp algorithm. These steps are described algorithmically by:

```

for  $k = 1$  to 2 (initial and final time)
     $\mathbf{A}_k = \mathbf{R}_k [\boldsymbol{\sigma}_k] \mathbf{R}_k^T$ 
    if  $k = 2$ 
        then perform rotation check on  $\mathbf{R}_2$ 
    end
     $\mathbf{R}_k \rightarrow [\theta_k, \vec{e}]$ 
     $\mathbf{q}_k = \left[ \cos\left(\frac{\theta_k}{2}\right), \vec{e}_1 \cdot \sin\left(\frac{\theta_k}{2}\right), \vec{e}_2 \cdot \sin\left(\frac{\theta_k}{2}\right), \vec{e}_3 \cdot \sin\left(\frac{\theta_k}{2}\right) \right]^T$ 
     $\mathbf{r}_k = \frac{1}{\sqrt{\text{diag } \boldsymbol{\sigma}_k}}$ , where  $\mathbf{r}_k$  is principal axes radii
end

```

Before interpolating we perform the following check on \mathbf{q}_2 to ensure the orientation is aligned within ± 90 degrees of \mathbf{q}_1 .

```

if  $\mathbf{q}_1 \cdot \mathbf{q}_2 < 0$ 
    then  $\mathbf{q}_2 = -\mathbf{q}_2$ 
end

```

Knowing the initial and final orientations, we can now calculate an ellipsoid at any intermediate time u that results in the minimum rotation angle using the Slerp algorithm, which appears as,

$$\theta_u = \cos^{-1}(\mathbf{q}_1 \cdot \mathbf{q}_2)$$

$$\mathbf{q}_u = \frac{\sin(1-u)\theta_u}{\sin(\theta_u)}\mathbf{q}_1 + \frac{\sin u\theta_u}{\sin(\theta_u)}\mathbf{q}_2$$

$$\mathbf{r}_u = \frac{\sin(1-u)\theta_u}{\sin(\theta_u)}\mathbf{r}_1 + \frac{\sin u\theta_u}{\sin(\theta_u)}\mathbf{r}_2$$

$$\mathbf{c}_u = \frac{\sin(1-u)\theta_u}{\sin(\theta_u)}\mathbf{c}_1 + \frac{\sin u\theta_u}{\sin(\theta_u)}\mathbf{c}_2$$

return $[\mathbf{q}_u, \mathbf{r}_u, \mathbf{c}_u]$

Continuing the example shown in Figure 5.4, we now apply this Slerp interpolation algorithm to identify the ellipsoid at time = 9.5 seconds. These results appear in Figure 5.7.

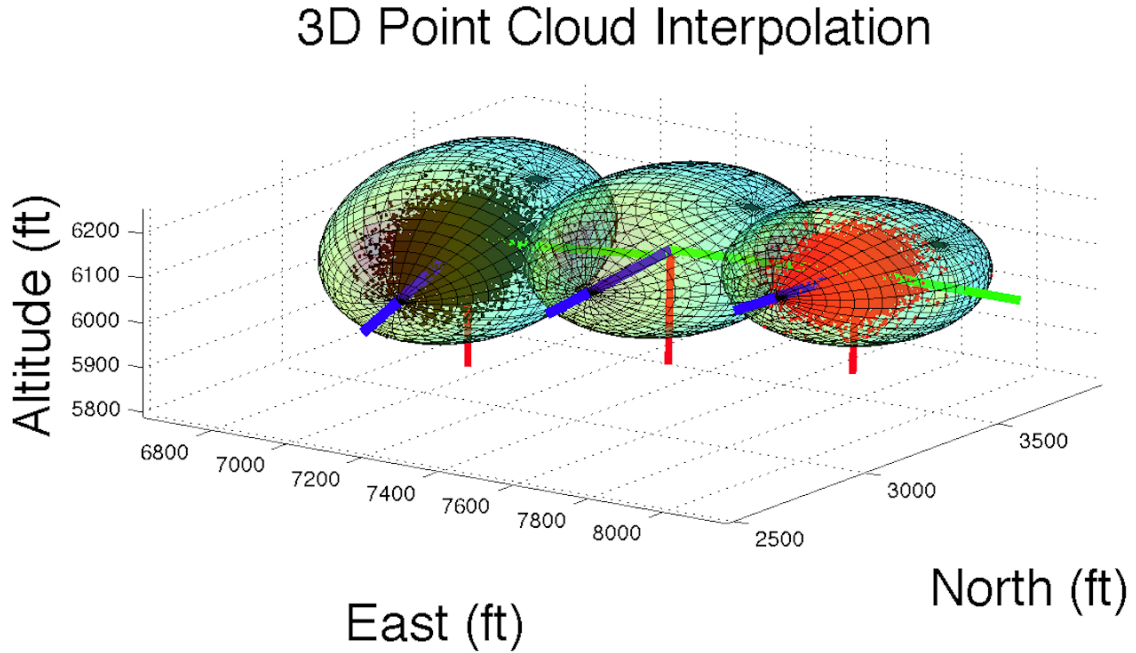


Figure 5.7: Slerp Interpolation at 9.5 Seconds

From this figure, we see that the algorithm can efficiently identify at any arbitrary time an ellipsoid representing the distribution for an intruder's position, and hence, can efficiently model a keep-out corridor for the optimal control problem. Further, since this keep-out corridor is generated from a sequential Monte Carlo process based on the best nonlinear estimate of the intruder's states, including turn-rate, this corridor should most accurately represent the inequality path constraint that is the most likely trajectory for the nonlinear airborne SAA optimal control problem. Thus, the SLIMVEE algorithm allows the optimal control problem to efficiently specify at any arbitrary time the boundaries of the keep-out corridor and provides an efficient check if trajectories violate the corridor.

5.5 Optimal Control Problem

5.5.1 Problem Formulation.

The formulation of the optimal control problem was described in Section 3.2. The performance measure we use in this chapter is to minimize state deviations from a time-varying path $\mathbf{C}(t)$ with minimum control effort as shown in equation (3.8) and repeated here for convenience:

$$J = \frac{1}{2} \int_{t_0}^{t_f} \left[(\mathbf{x} - \mathbf{C})^T \mathbf{Q} (\mathbf{x} - \mathbf{C}) + (\mathbf{u}^T \mathbf{R}_o \mathbf{u}) \right] dt, \quad \mathbf{Q} \geq 0 \quad \mathbf{R}_o > 0 \quad (5.34)$$

In this formulation our inequality path constraints, equation (3.5), are the interpolated ellipsoids representing the intruder's possible location at the specified optimization time step such that,

$$1 - [(\mathbf{x}_i - \mathbf{c})^T \mathbf{A} (\mathbf{x}_i - \mathbf{c})] \leq 0 \quad \text{where } i = 1, 2, 3 \quad (5.35)$$

where \mathbf{A} is the matrix containing the interpolated ellipsoid's magnitude and rotation angle and \mathbf{c} is the center of the interpolated ellipsoid.

In this chapter, the ownship attempts to minimize the deviation distance (d) from its planned trajectory between waypoints. In Figure 5.8 adapted from [5], \mathbf{x}_1 and \mathbf{x}_2 identify the start and end waypoints, \mathbf{x}_0 the current ownship position, and (d) the deviation distance such that [5]

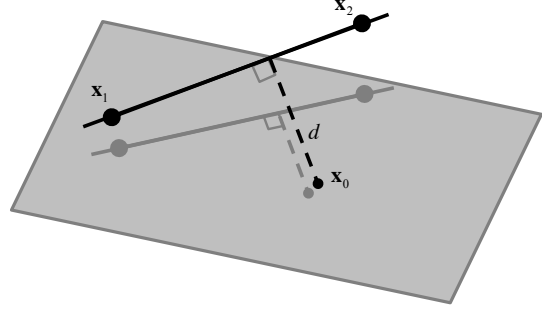


Figure 5.8: Point Line Distance from [5].

$$d = \frac{|(\mathbf{x}_0 - \mathbf{x}_1) \times (\mathbf{x}_0 - \mathbf{x}_2)|}{|\mathbf{x}_2 - \mathbf{x}_1|} \quad (5.36)$$

This chapter uses the commercial software package *GPOPS III* [78] to transcribe and solve the nonlinear optimal control problem.

5.5.2 Stochastic 3D Scenario Description.

The application for this collision avoidance algorithm is for NAS flight operations where aircraft usually do not perform aggressive maneuvers. The 8-state model described by equations (5.15) - (5.17) supports this type of flight operations and is the intruder model used in this stochastic 3D scenario. In this scenario the intruder flies at a constant speed of 290 ft/sec in standard-rate 3 deg/sec left turn while maintaining a shallow 1 deg flightpath angle (γ) climb. Equation (2.3) from Chapter II describes the ownship dynamics. In this scenario the ownship flies at a constant speed of 400 ft/sec and maneuvers with a maximum bank angle (μ) of ± 45 deg and a normal acceleration (N_z) of 0.59 to 1.41 g. Prior to the start of the simulation, we generate the intruder “truth trajectory” using the ownship dynamic equations. We do this for two reasons: First, these equations adequately model a deterministic 3D aircraft flight trajectory, and second, these equations allow us to better validate the performance of the estimation algorithm since we are using a different dynamics model to generate the “truth” than the model used by the estimation algorithm.

In this scenario, the optimization algorithm seeks to avoid the intruder while minimizing the deviation from a 3D flightpath corridor as described by equation (5.34), which generates an optimal 3D path for the ownship to fly that satisfies the boundary, path, and dynamic constraints. The simulation flies the ownship and intruder along the optimized trajectory

and the truth trajectory, respectively, until the next user-defined measurement time (1 Hz). As described earlier, the optimization algorithm uses a fixed 30-second receding horizon trajectory.

The estimation algorithm initializes by drawing 10,000 particles from a Gaussian distribution about a given a mean for each of the 8-states of the intruder model. The estimation algorithm then propagates all 10,000 particles every second for 30 seconds to produce an estimate of the intruder’s 8-state position, velocity, vertical acceleration, and turn-rate vector. The algorithm then calculates the MVEE at each 1 Hz propagation step and interpolates this result using the approach described in Section 5.3. This allows the algorithm to efficiently use the intruder’s time-varying probability region to specify the boundaries of the keep-out corridor for the inequality path constraint in equation (5.35) at each optimization evaluation time. Again, to account for additional uncertainty such as the physical dimensions of the intruder aircraft, we set the minimum keep-out corridor radius to 500 feet. Based on a RHC approach, the algorithm then performs a measurement update every 1 Hz and calculates a new 30-second receding horizon trajectory based on this update. As a result, the intruder’s probability region and associated keep-out corridor contracts and grows as a function of measurement update rate and time. Finally, to generate the optimal collision avoidance solution, we specify 30 fixed collocation nodes per receding horizon trajectory in the optimization software. We used 30 fixed collocation nodes for speed of solution; however, the method in this chapter is suitable for adaptive mesh schemes as available in GPOPS III.

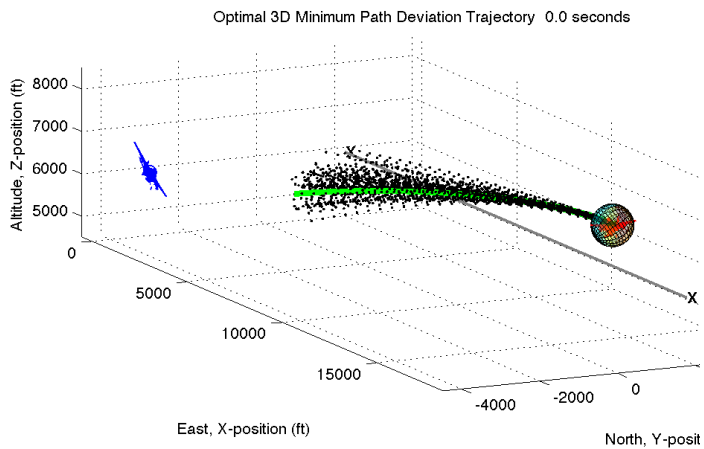
5.6 3D Stochastic Simulation Results

This section provides the 3D results of the simulation scenario described in the previous section. For this simulation, the ownship starts at an altitude of 6,000 feet and flies at a constant speed of 400 ft/sec in an easterly direction (positive x-axis). The ownship attempts to intercept the 3D flightpath corridor located co-altitude at 6,000 feet and parallel to the current flightpath but displaced laterally 5,000 feet to the north (y-axis). The intruder starts the simulation 10,000 feet east of the ownship and co-altitude at 6,000 feet heading

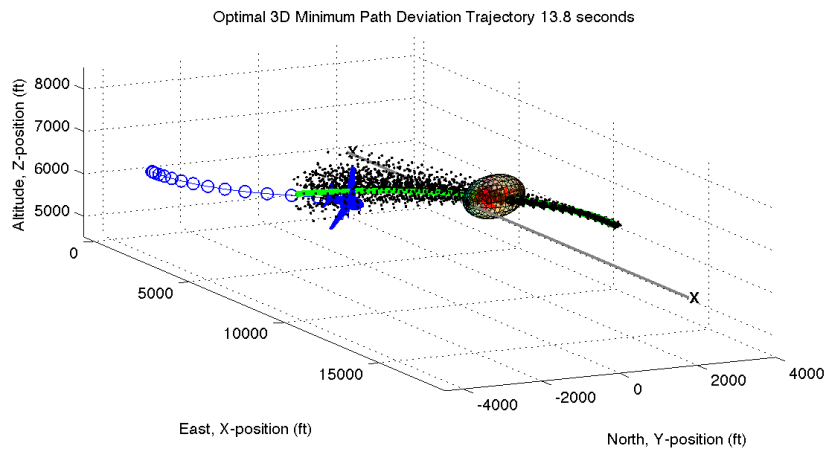
west (negative x-axis) with the 3D flightpath corridor displaced laterally 2,000 feet to the south (y-axis). The intruder then flies at a constant speed of 290 ft/sec in a standard-rate 3 deg/sec left turn while maintaining a shallow 1 deg flightpath angle (γ) climb. The initial separation distance between the two aircraft is approximately 12,200 feet.

For the simulation results presented, the blue aircraft represents the ownship and the red aircraft represents the intruder aircraft. The blue circles represent the calculated optimal avoidance trajectory at each time interval. The light-colored ellipsoid represents the keep-out corridor based on the interpolated point cloud distribution of the intruder's position estimate. The optimal control problem uses this keep-out corridor as an inequality path constraint in calculating an optimal avoidance trajectory. To assess system performance, the green line on the plot represents the "true" 3D intruder trajectory and the gray-colored line represents the ownship's intended 3D flightpath corridor. The black dots on the plot are the propagated particles estimate of the intruder's position at each measurement update (1 Hz). To prevent cluttering the plot, the results display only 1% of the actual particles.

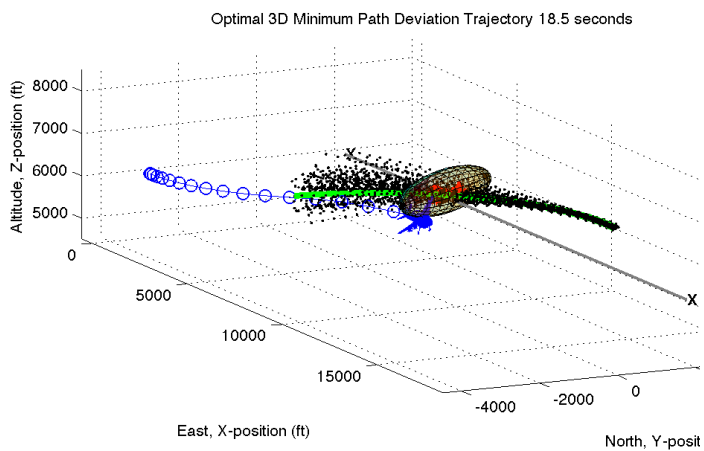
To establish a baseline for comparison and to demonstrate the feasibility of this approach, Figure 5.9 on the following page shows a time-sequenced quad chart of the algorithm's performance without any measurement updates throughout the entire 30-second time interval. In this scenario, the particle filter propagated the initial measurement distribution and calculated a 30-second 3D probability region estimate of the intruder. The optimization algorithm then generated a trajectory that avoided the keep-out corridor and minimized the deviation from the intended 3D flightpath (gray-colored line) as seen in Figure 5.9. In this scenario, at time zero, shown in panel (a) of Figure 5.9, the ownship began a level left bank turn to intercept the desired 3D flightpath corridor (gray-line). At approximately 13.8 seconds as seen in panel (b) of Figure 5.9, the ownship reversed the turn to smoothly intercept the 3D flightpath corridor (gray-line); however, as shown in panel (c) of Figure 5.9, at approximately 18.5 seconds the ownship reached the intruder's probability region and had to descend to avoid this keep-out corridor. Once the intruder's probability



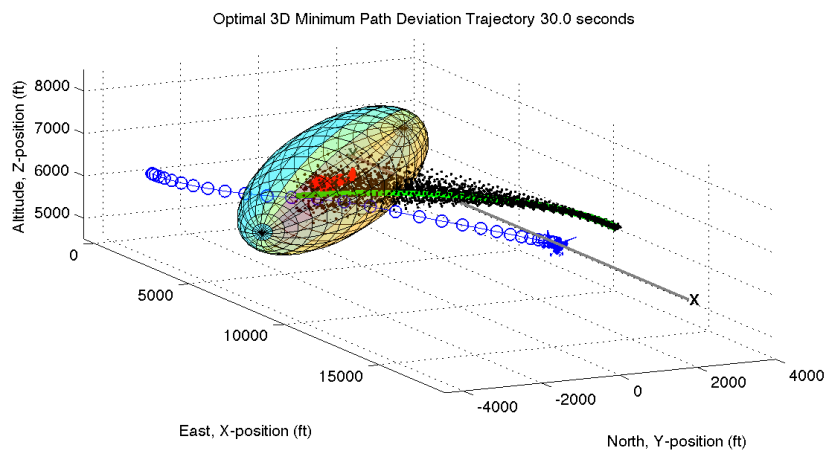
(a) Time 0 seconds



(b) Time 13.8 seconds



(c) Time 18.5 seconds



(d) Time 30 seconds

Figure 5.9: Time Series for Ownship (Blue) Avoiding the Intruder's (Red) Probability Region with No Measurement Updates.

region passed and was no longer an active constraint, the ownship began a shallow climb and continued to intercept the 3D flightpath corridor as shown in panel (d) of Figure 5.9.

For this baseline scenario without a measurement update, Figure 5.10 shows the separation distance between the ownship and the true intruder's position. In this figure, the gray-colored line at 500 feet represents the minimum radius of the keep-out corridor. The minimum separation distance between the ownship and the true intruder trajectory (green) was 942.7 feet and this occurred at 19 seconds in this scenario. Despite not having any measurement updates during the 30-second time horizon, the algorithm performed well. The optimal trajectory satisfied the inequality path constraints of avoiding the intruder's dynamically changing probability region while minimizing path deviations from the desired 3D flightpath corridor. However, in the absence of measurements, over time the filter's estimate of the intruder degrades. Receiving regular measurement updates improves the filter's estimate and reduces the size of the keep-out corridor.

Figure 5.11 on the next page shows the results of incorporating measurement updates. In general, the measurement updates reduce the intruder's probability region, and hence, the keep-out corridor, which subsequently causes the ownship to not deviate as far from the intended flightpath corridor in order to avoid the intruder. The scenario for these new results are identical to the previous scenario depicted in Figure 5.9 on the preceding page for both the ownship and intruder aircraft. The only exception is that now the algorithm includes regular measurement updates at a 1 Hz rate using the measurement model described earlier in Section 5.1. Figure 5.11 helps visualize the performance benefits of measurement updates by first showing the particle filter's initial estimate of the ellipsoids associated with the

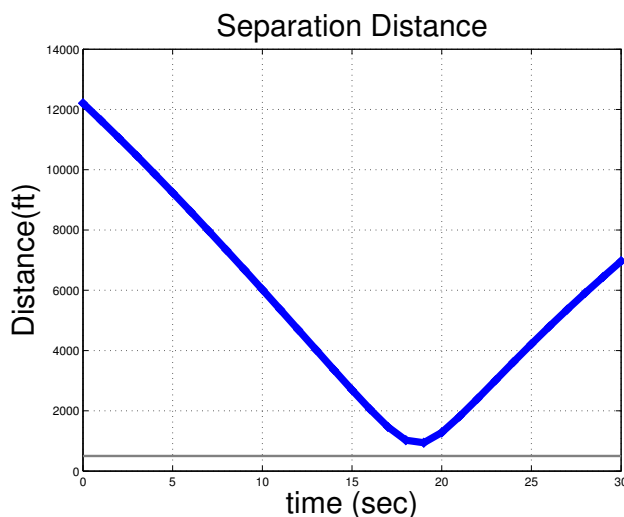
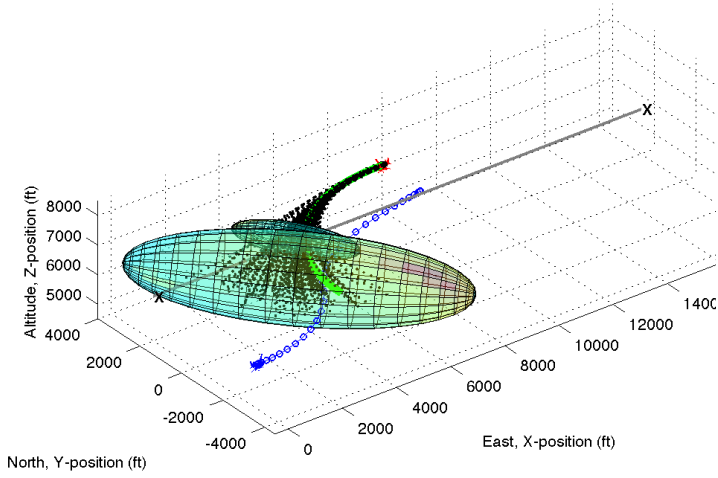


Figure 5.10: No Measurement Updates

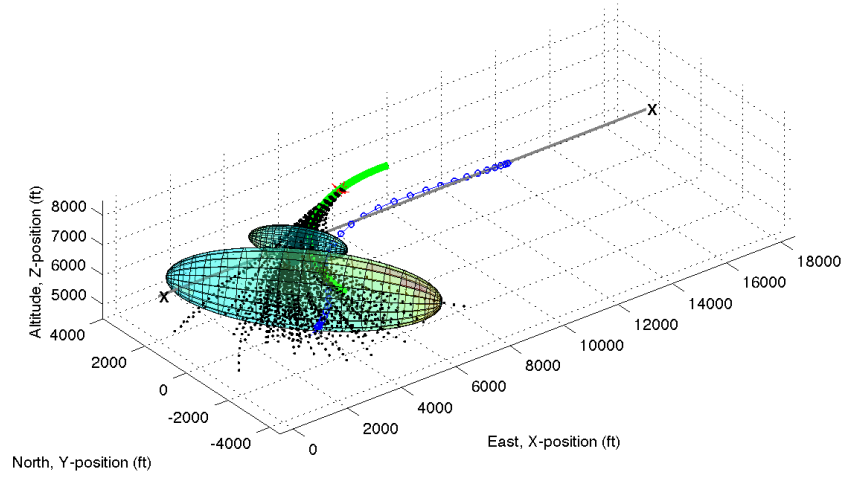
Figure 5.11 helps visualize the performance benefits of measurement updates by first showing the particle filter's initial estimate of the ellipsoids associated with the

Optimal 3D Minimum Path Deviation Trajectory 0.0 seconds



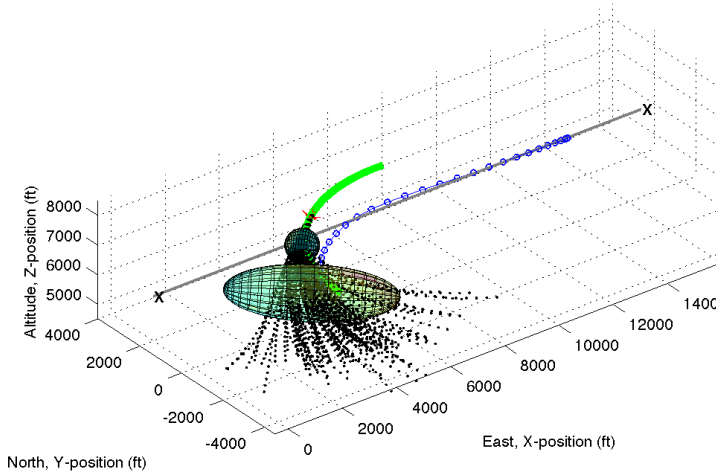
(a) Time 0 seconds

Optimal 3D Minimum Path Deviation Trajectory 7.0 seconds



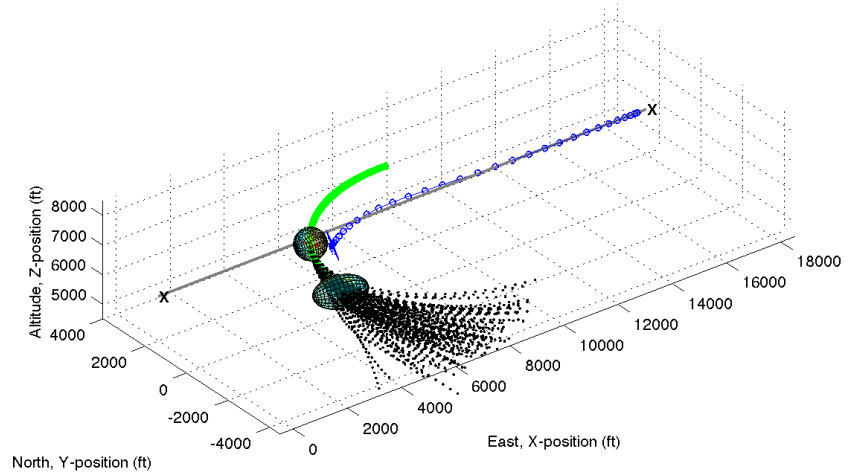
(b) Time 7 seconds

Optimal 3D Minimum Path Deviation Trajectory 13.0 seconds



(c) Time 13 seconds

Optimal 3D Minimum Path Deviation Trajectory 19.0 seconds



(d) Time 19 seconds

Figure 5.11: Time Series for Ownship (Blue) Avoiding the Intruder's (Red) Probability Region with Measurement Updates.

intruder's probability region at the 19 and 30-second point on the trajectory based on the initial measurement update at time zero. The simulation then incorporates regular measurement updates at a 1 Hz rate. Immediately following the measurement updates at 7, 13, and 19 seconds, separate time-sequence panels in Figure 5.11 display the true intruder and ownship position and show how the original probability regions associated with the 19 and 30-second ellipsoids shrink from the initial filter estimate generated at time zero. This comparison visually shows how the keep-out corridor decreases with increased number of measurement updates. To facilitate a better comparison of the change in ellipsoid size, Figure 5.11 is at a different view angle than Figure 5.9.

Panel (a) of Figure 5.11 visually shows the turn-rate FOGM model, equation (5.13), operated correctly. In this panel, the point cloud distribution initially closely followed the green truth trajectory based on the initial 3 deg/sec estimate of the turn-rate state; however, due to the 30-second maneuver time constant, the FOGM model correctly caused the turn-rate estimate, and subsequently the point cloud distribution, to transition back towards a zero-mean value over time due to the decreased probability in the validity of this initial estimate caused by the absence of regular measurement updates. Panels (b), (c) and (d) of Figure 5.11, visually highlight the benefits of measurements updates. The keep-out corridor boundaries associated with the 19 and 30-second probability region are noticeably smaller with increased measurement updates. In this scenario the closest point of approach (CPA) occurred at 19 seconds. As seen in panel (d) of Figure 5.11, the boundaries of the keep-out corridor at the CPA was at the minimum radius of 500 feet.

Figure 5.12 shows the separation distance between the ownship and the true intruder's position for each measurement time comparison shown in Figure 5.11. For additional insight, Figure 5.12 also shows the separation distance at two additional measurement start times of 17 and 20 seconds, which occur shortly prior and immediately after the CPA, respectively. In Figure 5.12, the blue line represents the separation distance for the initial trajectory without a

measurement update; this is the same plot as Figure 5.10. The red line in Figure 5.12 is the separation distance starting at time 7 seconds and the green line is the separation distance starting at 13 seconds. The light-blue or cyan-colored, the black, and the magenta lines are the separation distances starting at 17, 19, and 20 seconds, respectively. Again, the gray-colored line at 500 feet in Figure 5.12 represents the minimum radius of the keep-out corridor. Based on the regular measurement updates, the minimum separation distance between the ownship and the true intruder trajectory decreased to 661.7 feet at the CPA, which is nearly 300 feet closer than the no measurement update scenario.

5.7 Analysis of Results

The previous section demonstrated the validity of the stochastic intruder model and the approach of formulating the keep-out corridor as a dynamic inequality constraint for the nonlinear optimal control problem based on the intruder's estimated time-varying probability region. The algorithm correctly estimated the intruder's trajectory and defined at each evaluation time the boundaries of the intruder's probability region. The algorithm also correctly determined the optimal trajectory that minimized path deviations to the desired 3D flightpath corridor while avoiding the intruder. Using the modified turn rate model, the filter accurately estimated the intruder's turn rate and the FOGM model

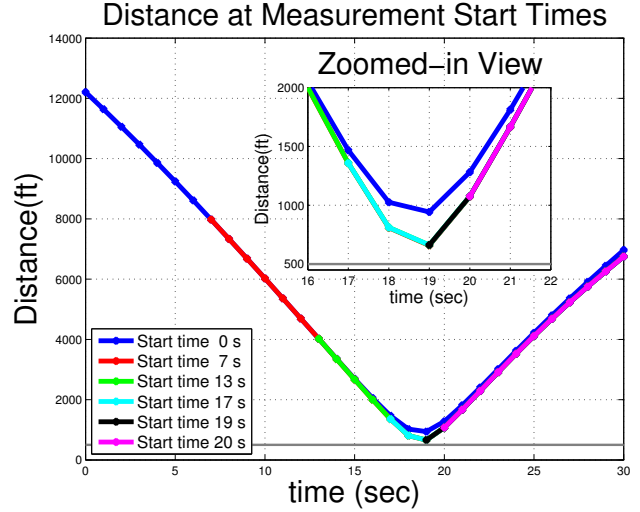


Figure 5.12: Separation Distance with Measurement Updates

correctly caused the turn-rate estimate to tend to a zero-mean estimate at the appropriate rate in the absence of measurements.

To further assess the performance of the optimization algorithm we repeated the simulation in the previous section with the identical initial conditions but removed the intruder from the scenario. As suspected, in this no-intruder simulation the ownship performed an immediate level left turn using maximum available controls and then performed a level right turn with maximum

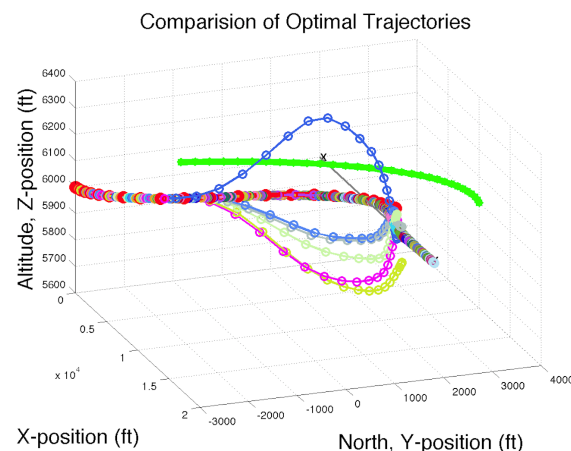


Figure 5.13: Optimal Path Comparison

available controls to quickly intercept the desired 3D flightpath. We then introduced the intruder back into the scenario and visually looked at how these trajectories compared as a function of measurement updates with the no-intruder optimal trajectory. Figure 5.13 shows these results. In this figure the no-intruder trajectory is highlighted in red and intruder avoidance trajectories are in different colors based on the measurement start time. Also in this figure the true intruder trajectory is highlighted in green and the gray-colored line shows the desired 3D flightpath. The axes in this figure are intentionally unequal since the majority of the deviation from the nominal path occur in the vertical direction making these deviation difficult to see with equal axes. Prior to reaching the intruder's probability region, each path follows the red baseline trajectory. Once this inequality constraint becomes active, that is the ownship reaches the intruder's probability region, the algorithm then deviates from the nominal no-intruder path and climbs or descends as indicated on Figure 5.13. This figure is only intended to show that in general, the presence of the intruder caused the optimization algorithm to deviate from the nominal optimal path. Further, continued measurement updates allowed the particle filter to get a better estimate of the intruder and reduce the probability region surrounding the intruder. As a result, the optimization

algorithm then could make smaller deviations away from the nominal no-intruder trajectory, and hence, minimize the overall deviation from the desired 3D flightpath.

Although not a focus of this research, another area worthy of mention is algorithm execution time. An often-stated critique of global path planning methods such as an optimal control problem formulation solved using pseudospectral or direct orthogonal collocation is processing speed [4]. The work in this chapter was not intended to demonstrate real-time operations of an airborne collision avoidance system, the focus was to show the methodology and considerations for modeling and incorporating constraints for the optimal control problem in a stochastic environment; nevertheless, researchers have demonstrated direct orthogonal collocation methods in flight test with an unmanned aircraft for path following applications [38].

We ran the simulations in this chapter using Matlab[®] version 2012b on a laptop computer operating with OS X version 10.9 operating system and a 2.3 GHz Intel Core *i5* processor with 16 GB 1333 MHz DDR3 memory. In this chapter the simulation algorithms were not necessarily optimized for speed but were coded for robust post-processing analysis. Using parallel processing or a compiled language such as C++ would greatly reduce processing time. Nonetheless, as a initial gauge of performance speeds, the optimization algorithm in this chapter took approximately 6 to 8 seconds to calculate an optimal 30-second trajectory and the SLIMVEE algorithm took approximately 2.5 seconds per 30-second time trajectory. The largest amount of time was spent on the particle filter algorithm. This was largely due to propagating 10,000 particles every second for 30 seconds to estimate each of the 8 intruder states. (Chapter VIII analyzes filter performance and efficiency for different number of particles.) In general, the particle filter algorithm took approximately 18 seconds to perform the measurement update, resample, and then generate a new 30-second estimated trajectory; again, this algorithm was not optimized for speed. Due to the uncorrelated nature of the particles, for real-time implementation the processing time of the particle propagation could be reduced greatly through parallelization by software or application specific integrated circuits. Another key consideration, as noted [44], for

any particle filter implementation is to employ a “Rao-Blackwellization marginalization”, if possible, to allow the particle filter to operate with a much lower dimensionality to help increase efficiency. Processing time reduction through this marginalization is a potential area for future research.

Another significant area of concern for a real-time collision avoidance implementation using a direct method is the failure of the NLP solver to find a feasible solution. Lai and Whidborne [90] applied a direct collocation method to solve the optimal control problem for an unmanned obstacle avoidance application. In their formulation, they included an “onboard backup trajectory...to handle the possible not-converged situations” [90]. In an optimal airborne collision avoidance application, a failure to converge is a critical area that requires additional study to better understand root causes for potential infeasible solutions, and how best to mitigate against these contingencies in a real-time implementation.

Finally, an important consideration in any optimal control problem formulation is on how to define the performance measure or cost function. In this scenario, the cost function was simply a balance between minimizing path deviation and control usage. However, in a real-world NAS application the cost function should probably be more complex. For example, in the scenario in this chapter the optimization algorithm correctly calculated the optimal avoidance trajectory that minimized the path deviation; however, at the CPA the ownship, although offset slightly in altitude, nearly crossed directly in front of the intruder’s intended flightpath as seen in Figure 5.14.

Note in this figure the diameter of the sphere surrounding the intruder aircraft (red) is 1,000 feet but the ownship (blue) and intruder are shown larger than scale. Although the trajectory in Figure 5.14 produced the lowest cost and was optimal as defined by the performance measure, this

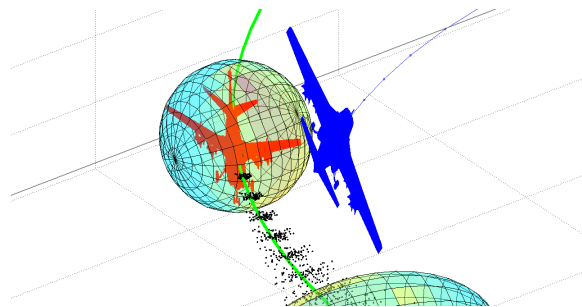


Figure 5.14: Closest Point of Approach

path may have violated standard rules of aviation by turning directly in front of an

approaching aircraft. Therefore, a further area of research to explore is defining appropriate cost functions for the optimal control problem for operations in the NAS which incorporate Federal Aviation Regulations ‘rules of the road.’

5.8 Conclusion

This chapter showed a general framework for stochastic intruder modeling and constraint formulation when posing the airborne SAA as a nonlinear optimal control problem. The approach in this chapter did not rely on a Gaussian distribution, but instead used a 3D particle filter to estimate an intruder’s probability regions as ellipsoids and then smoothly interpolated these ellipsoids to accurately identify a time-varying collision avoidance corridor used as a path constraint for the nonlinear optimal control problem. This approach was applied on a representative scenario and gave favorable results. As described earlier, there are many advantages for using an optimal control approach; however, the potential limitations of this approach are having sufficient onboard processing to implement in real-time and handling situations when the NLP solver fails to find a feasible solution. Future steps in this research effort include validating the robustness of the model, comparing the performance of the particle filter against other nonlinear filters such as the EKF and UKF, and determining performance advantages of different stochastic estimation techniques as well as investigating alternative formulations of the cost function for “real-world” scenarios. The next step is to look at formulating conditional constraints which are imposed when examining the real-world problem such as obeying rules-of-the-road and FAA regulations.

VI. Implementing Conditional Inequality Constraints for Optimal Collision Avoidance

CURRENT FEDERAL Aviation Administration regulations require that passing aircraft must either meet a specified horizontal *or* vertical separation distance. However, solving for optimal avoidance trajectories with conditional inequality path constraints is problematic for gradient-based numerical nonlinear programming solvers since conditional constraints typically possess non-differentiable points. Further, the literature is silent on robust treatment of approximation methods to implement conditional inequality path constraints for gradient-based numerical nonlinear programming solvers. This chapter⁵ proposes two efficient methods to enforce conditional inequality path constraints in the optimal control problem formulation and compares and contrasts these approaches on representative airborne avoidance scenarios. The first approach is based on a minimum area enclosing superellipse function and the second is based on use of sigmoid functions. These proposed methods are not only robust, but also conservative, that is, their construction is such that the approximate feasible region is a subset of the true feasible region. Further, these methods admit analytically derived bounds for the over-estimation of the infeasible region with a demonstrated maximum error of no greater than 0.3% using the superellipse method, which is less than the resolution of typical sensors used to calculate aircraft position or altitude. However, the superellipse method is not practical in all cases to enforce conditional inequality path constraints that may arise in the nonlinear airborne collision avoidance problem. Therefore, this chapter also highlights by example when the use of sigmoid functions are more appropriate.

6.1 Introduction

Under the Federal Aviation Administration (FAA) Modernization and Reform Act of 2012, the United States Congress tasked the FAA to “provide for the safe integration of

⁵Note: This entire chapter is a collaborative work with Maj Christopher Arendt intended for archival journal submission [91]. This chapter cites in the appropriate sections unique contributions by Maj Arendt.

civil unmanned aircraft systems into the national airspace system as soon as practicable, but not later than September 30, 2015” [13]. A means to meet this integration mandate is through the use of algorithms that autonomously generate optimal collision avoidance trajectories to satisfy current FAA regulations that mandate passing aircraft meet either a minimum horizontal *or* vertical separation distance. A number of works have looked at trajectory planning and optimization for air vehicles using optimal control problem formulations [35–37] and some, such as [38], have even demonstrated this method in flight on a small-size unmanned vehicle. However, a potential limitation of this method is enforcing conditional inequality constraints such as maintaining either a minimum horizontal or vertical separation distance from an approaching aircraft or complying with FAA right of way (ROW) rules. For example, according to FAR 91.113 if two aircraft are approaching nearly head on, then “each aircraft shall alter course to the right.” Optimal control problems are often solved using gradient-based numerical nonlinear programming (NLP) solvers which require smooth differentiable constraints; however, conditional constraints are not always differentiable, and thus can cause gradient-based numerical solvers to fail. This chapter proposes and analyzes two different methods to address the issue of non-differentiable conditional inequality path constraints. The first approach is based on a minimum area enclosing superellipse (MAES) function and the second is based on the use of sigmoid functions. Both of these approaches are differentiable, allowing the NLP solver to calculate gradients and find an optimal solution.

Standard methods for implementing conditional inequality constraints can be classified as indicator methods [92], including Big M [93, 94] and active set [95] methods, and mixed-norm methods [92, 96]; however, these methods are not everywhere-differentiable, and therefore, they often cause gradient-based NLP solvers to fail to generate an optimal solution [92]. For instance, Big M methods implement “either-or constraints” [94] using a binary indicator variable along with a sufficiently large constraint variable (M); thus, constraints become non-differentiable with respect to the binary indicator variable [92]. Similarly, active set methods use the conditional constraints to define sub-sets of feasible

solutions and then optimize over each sub-set when it is indicated as the active set [92]. However, these methods are not well suited for dynamic conditional constraints such as the time varying airborne collision avoidance problem since the continuous-time constraints implicitly define an uncountable number of feasible sub-sets, while discretizing the constraints introduces an implicit or explicit binary indicator variable equivalent to those in Big M methods [95]. In addition, mixed-norm methods typically formulate a set of conditional constraints as a single constraint involving the maximum of a set of norms from each conditional constraint [92, 96]; thus, the constraint’s derivative at a point is a function of the derivative of the norm that obtains the maximum value at that point [92]. Therefore, if the mixed set of norms do not have identical derivatives at points where the maximum norm changes from one norm to another in the set, the mixed-norm formulation will not be everywhere-differentiable [92]. In the context of collision avoidance, [35] devised a novel approach for enforcing a conditional inequality constraint of maintaining either a minimum horizontal or vertical separation distance from an approaching aircraft; however, their approach did not address situations with more than two conditional constraints and required the introduction of an additional control variable appended to the objective function. An approach that is similar to the methods in this chapter is known as artificial potential fields or functions, or APF. While APF methods are differentiable [28, 97], they do not truly enforce conditional inequality constraints [92]. Instead, APF methods treat path constraints as “soft” obstacles and incorporate them as weighted penalties in the cost function which may result in generating infeasible trajectories [92, 97]. However, the methods proposed in this chapter provide conservative and differentiable approximations for indicator methods as well as mixed-norm methods [92], thus ensuring differentiability for the gradient-based NLP solver while maintaining feasibility for the optimal control problem.

The overview of this chapter is as follows: Section 6.2 introduces and develops the MAES and sigmoid conditional constraint approximation methods. Sections 6.5 - 6.12 describe and then analyze the simulation results from the two example problems in this chapter and Section 6.14 summarizes the results.

6.2 Conditional Inequality Constraint Approximation Methods

This section begins by introducing the first of the two example problems in this chapter to properly motivate the development of the conditional constraint formulation methods presented herein. In the first example problem, the objective is for the ownship to minimize deviations from a 3D flight path corridor while maintaining either a horizontal separation distance (Δxy) of at least 2460 ft *or* a vertical separation distance (Δz) of at least 820 ft from an intruder aircraft where: ⁶

$$\Delta xy = \sqrt{(x_{\text{intruder}} - x_{\text{ownship}})^2 + (y_{\text{intruder}} - y_{\text{ownship}})^2} \quad (6.1)$$

$$\Delta z = |z_{\text{intruder}} - z_{\text{ownship}}| \quad (6.2)$$

Using logic '*if statements*', this inequality constraint formulation appears algorithmically as:

$$\begin{aligned} &\text{for each collocation node } i = 1 \text{ to } n \\ &\quad \text{if } \Delta xy(i) > 2460 \\ &\quad \quad h_{\text{sep}}(i) = 1 \\ &\quad \text{end} \\ &\quad \text{if } \Delta z(i) > 820 \\ &\quad \quad v_{\text{sep}}(i) = 1 \\ &\quad \text{end} \\ &\text{end} \\ &1 - [h_{\text{sep}} + v_{\text{sep}}] \leq 0 \text{ **inequality path constraint**, } \in \mathbb{R}^n \end{aligned} \quad (6.3)$$

This definition for the conditional inequality path constraint in equation (6.3) was evaluated using two different gradient-based NLP solvers, IPOPT and SNOPT; however, as expected, due to the non-differentiable conditional constraint caused by the logic '*if statements*', both solvers failed to converge to a solution since they could not determine a gradient direction to search in order to find an extremal point. Therefore, an alternate approach is needed to enforce a conditional constraint without the use of logic '*if statements*'. The two approaches

⁶The distances 820 ft (250 m) and 2460 ft (750 m) are assumed as initial planning guidance for developing avoidance algorithms to support the integration of remotely piloted aircraft into the NAS.

analyzed in this chapter are (1) MAES and (2) sigmoid functions to approximate inequality path constraints for the optimal control problem. The next sections describe these two approaches.

6.3 Minimum Area Enclosing Superellipse (MAES)

The equation for a superellipse appears as:

$$\left(\frac{x}{a}\right)^N + \left(\frac{y}{b}\right)^N = 1 \quad (6.4)$$

where a and b represent the semi-major and semi-minor axes of the superellipse while $N \geq 2$ is an even number [98]. The equation for the area of a superellipse [99] appears as:

$$Area = 4abC(N) \quad (6.5)$$

where $C(N)$ is a ratio of gamma functions of N defined as [99]:

$$C(N) = \frac{(\Gamma(1 + \frac{1}{N}))^2}{\Gamma(1 + \frac{2}{N})} \quad (6.6)$$

Additionally, a superellipse that encloses a rectangle with length $(2h)$ and width $(2v)$ centered at the origin must intersect each of the 4 corners of the rectangle. That is, the minimum area enclosing superellipse must satisfy,

$$\begin{aligned} \left(\frac{+h}{a}\right)^N + \left(\frac{+v}{b}\right)^N &= 1 \\ \left(\frac{-h}{a}\right)^N + \left(\frac{+v}{b}\right)^N &= 1 \\ \left(\frac{+h}{a}\right)^N + \left(\frac{-v}{b}\right)^N &= 1 \\ \left(\frac{-h}{a}\right)^N + \left(\frac{-v}{b}\right)^N &= 1 \end{aligned} \quad (6.7)$$

Since N must always be an even number, all 4 constraint expressions in equation (6.7) are equivalent. Furthermore, since $4C(N)$ in equation (6.5) is not a function of a or b , the optimization problem to determine the semi-major and semi-minor axes values, a^* and b^* respectively, that minimize the area of an enclosing superellipse for any even $N \geq 2$ reduces to:

$$\begin{aligned} &\textbf{minimize } ab \\ &\textbf{such that } \left(\frac{h}{a}\right)^N + \left(\frac{v}{b}\right)^N = 1, \text{ where } a, b \geq 0 \end{aligned}$$

It is easy to verify that the values $a^* = 2^{1/N}h$ and $b^* = 2^{1/N}v$ satisfy the first-order KKT and second-order sufficiency conditions for optimality. Therefore, the equation of the minimum area superellipse that encloses the minimum separation rectangle appears as:

$$\left(\frac{x}{h}\right)^N + \left(\frac{y}{v}\right)^N = 2 \quad (6.8)$$

where h and v represent the minimum horizontal and vertical separation distance constraint, respectively.

Raising the exponential term, N , in equation (6.8) to higher-order even powers causes the superellipse to appear increasingly rectangular. Figure 6.1 graphically shows the results of increasing the exponential terms in equation (6.8). In this figure, the x-axis represents the horizontal separation constraint (Δxy) and the y-axis the vertical separation constraint (Δz). The red dashed lines depict the minimum separation distance of ± 2460 ft and ± 820 ft in the horizontal (h) and vertical (v), respectively. From Figure 6.1, in the limit as $N \rightarrow \infty$ the superellipse approaches the rectangular conditional constraints. Substituting x and y in equation (6.8) with Δxy and Δz , respectively, gives the superellipse equation as:

$$\left(\frac{\Delta xy}{h}\right)^N + \left(\frac{\Delta z}{v}\right)^N = 2 \quad (6.9)$$

Note: Equations 6.10 - 6.19 and the associated text are unique contributions by Maj Arendt that do not appear in [92] and only appear below.

Note that,

$$\left(\frac{\Delta xy}{h}\right)^N + \left(\frac{\Delta z}{v}\right)^N = 2 \iff \lim_{N \rightarrow \infty} \left(\left(\frac{\Delta xy}{h}\right)^N + \left(\frac{\Delta z}{v}\right)^N \right)^{\frac{1}{N}} = \lim_{N \rightarrow \infty} 2^{\frac{1}{N}} \quad (6.10)$$

Furthermore [100],

$$\lim_{N \rightarrow \infty} \left(\left(\frac{\Delta xy}{h}\right)^N + \left(\frac{\Delta z}{v}\right)^N \right)^{\frac{1}{N}} \triangleq \max \left\{ \frac{\Delta xy}{h}, \frac{\Delta z}{v} \right\} \quad (6.11)$$

Therefore, if the mixed-norm is defined as [96]:

$$\left\| \left(\frac{\Delta xy}{h}, \frac{\Delta z}{v} \right) \right\|_{\text{mixed}} \triangleq \max \left\{ \frac{\Delta xy}{h}, \frac{\Delta z}{v} \right\} \quad (6.12)$$

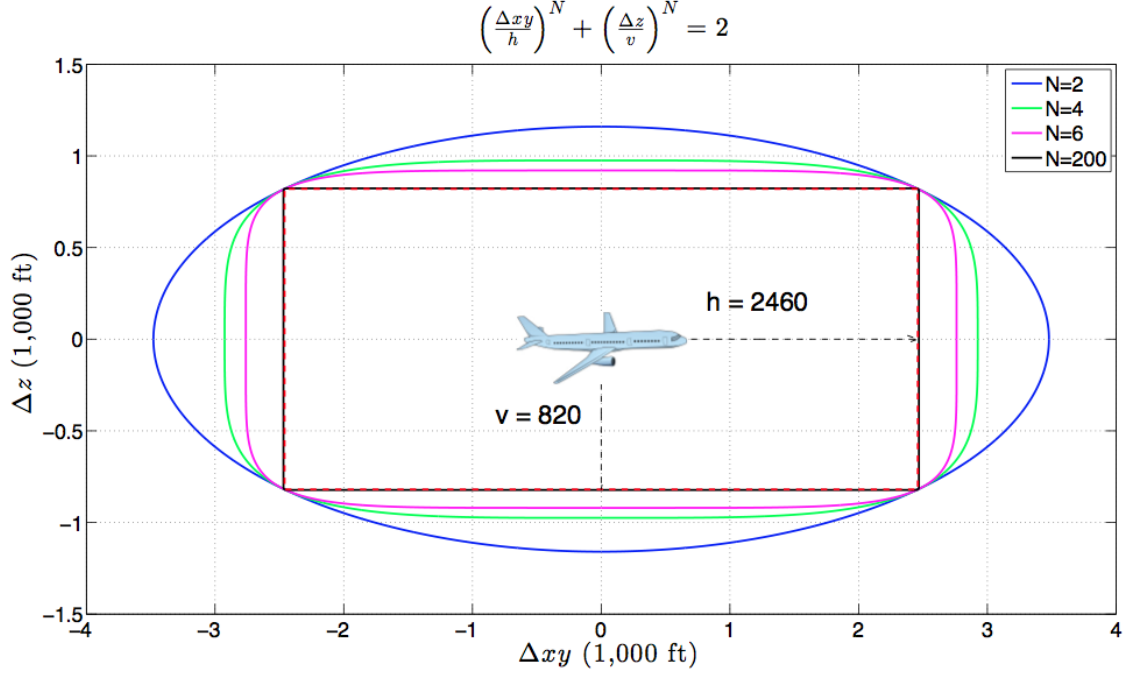


Figure 6.1: Approximating Conditional Inequality Path Constraints

then in the limit as $N \rightarrow \infty$ the superellipse equation (6.9) is equivalent to the dashed-red rectangle in Figure 6.1 given by the mixed-norm equation:

$$\left\| \left(\frac{\Delta xy}{h}, \frac{\Delta z}{v} \right) \right\|_{\text{mixed}} = 1 \quad (6.13)$$

since,

$$\max \left\{ \frac{\Delta xy}{h}, \frac{\Delta z}{v} \right\} = 1 \iff \lim_{N \rightarrow \infty} \left(\left(\frac{\Delta xy}{h} \right)^N + \left(\frac{\Delta z}{v} \right)^N \right)^{\frac{1}{N}} = \lim_{N \rightarrow \infty} 2^{\frac{1}{N}} = 1 \quad (6.14)$$

An advantage of using a MAES to approximate the conditional inequality constraint is that since the semi-major and semi-minor axes are defined as $a^* = 2^{\frac{1}{N}} h$ and $b^* = 2^{\frac{1}{N}} v$, respectively, the overestimation errors (δ_h and δ_v) for infeasible values of Δxy and Δz are bounded such that:

$$\begin{aligned} 0 \leq \delta_h &\leq \left(2^{1/N} - 1 \right) h \\ 0 \leq \delta_v &\leq \left(2^{1/N} - 1 \right) v \end{aligned} \quad (6.15)$$

From equation (6.15), for given h and v the MAES overestimation error is strictly a function of N . For the example in this chapter, the MAES method used a value of $N = 200$, resulting

in a maximum overestimation error of 0.3%, which is less than the typical resolution of an aircraft's onboard sensors used to calculate position or altitude. Therefore, system designers should select the appropriate value of N based on sensor resolution. The minimum value of N necessary to guarantee the overestimation error is less than the sensor tolerance, Δ_s , is given by the smallest even number that satisfies the following relationships:

$$\begin{aligned} N &\geq \frac{\ln 2}{\ln \left(\frac{\Delta_s}{h} + 1 \right)} \\ N &\geq \frac{\ln 2}{\ln \left(\frac{\Delta_s}{v} + 1 \right)} \end{aligned} \quad (6.16)$$

For example, if $\Delta_s = 12$ feet while $h = 2460$ feet and $v = 820$ feet, then:

$$\begin{aligned} N &\geq \frac{\ln 2}{\ln \left(\frac{12}{2460} + 1 \right)} = 142.44 \\ N &\geq \frac{\ln 2}{\ln \left(\frac{12}{820} + 1 \right)} = 47.71 \end{aligned} \quad (6.17)$$

Therefore, N would be set to 144.

However, for large values of N , constraints involving the equation of the superellipse become computationally difficult to evaluate. This issue of *constraint scaling* is addressed by applying the natural log on equation (6.8) to generate the equivalent equation of the minimizing enclosing superellipse. The adapted equation appear as:

$$\ln \left(\left(\frac{\Delta xy}{2460} \right)^N + \left(\frac{\Delta z}{820} \right)^N \right) = \ln 2 \quad (6.18)$$

Therefore, the standard form of the inequality path constraint for the optimal control problem appears as:

$$\ln 2 - \ln \left(\left(\frac{\Delta xy}{2460} \right)^N + \left(\frac{\Delta z}{820} \right)^N \right) \leq 0 \quad (6.19)$$

Equation (6.19) is the form of the MAES method used with equations (6.1) and (6.2) at each collocation node to solve the first example problem in this chapter. However, it may be impractical to apply the MAES method to optimal control problems with multiple, compound (or *nested*) conditional inequality constraints, represented by the second example problem. To address this limitation, the next section develops differentiable approximations of indicator methods using a sigmoid function form of the conditional inequality path constraint.

6.4 Sigmoid Function

Another approach to incorporate a conditional constraint is to use a sigmoid function approximation of a conditional indicator function [92]. The earlier section described the problem that gradient-based NLP solvers have with non-differentiable conditional constraints. Like the MAES, sigmoid functions avoid this problem since they too are continuous and differentiable. Framing the development in the context of the first example problem, two unique sigmoid functions are defined to approximate the horizontal and vertical inequality path constraint indicator functions separately. The equations for the horizontal and vertical sigmoid functions, S_h and S_v , that approximate the inequality path constraint indicator functions appear as [92]:

$$S_h(\Delta xy, s_h) = \left[1 + e^{s_h \left(1 - \frac{\Delta xy}{2460} \right)} \right]^{-1} \quad (6.20)$$

$$S_v(\Delta z, s_v) = \left[1 + e^{s_v \left(1 - \frac{\Delta z}{820} \right)} \right]^{-1} \quad (6.21)$$

where s_h and s_v are user-defined positive stiffness factors for the smoothness and orientation of their respective sigmoid. Figure 6.2 shows the results of plotting equations (6.20)

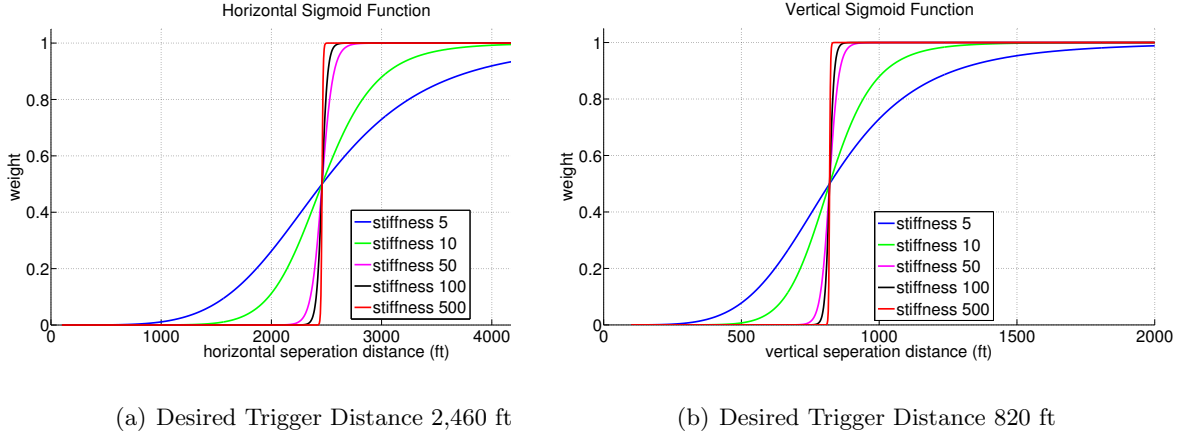


Figure 6.2: Horizontal and Vertical Separation Sigmoid Functions

and (6.21) for varying values of positive s_h and s_v . Negative values of s_h and s_v merely reflect the image of the sigmoid function about the critical values (2460 and 820). From

Figure 6.2, when $\Delta xy = 2460$ or $\Delta z = 820$ the value of the respective sigmoid equals 0.5. When $\Delta xy < 2460$ or $\Delta z < 820$ then the value of the respective sigmoid approaches zero. Likewise, when $\Delta xy > 2460$ or $\Delta z > 820$, the value of the respective sigmoid approaches unity. Therefore, the sigmoid functions define the inequality constraint indicator function approximations which appear as [92]:

$$S_h(\Delta xy, s_h > 0) \approx \begin{cases} 1, & \Delta xy \geq 2460 \\ 0, & \text{otherwise} \end{cases} \quad S_h(\Delta xy, s_h < 0) \approx \begin{cases} 0, & \Delta xy \geq 2460 \\ 1, & \text{otherwise} \end{cases} \quad (6.22)$$

$$S_v(\Delta z, s_v > 0) \approx \begin{cases} 1, & \Delta z \geq 820 \\ 0, & \text{otherwise} \end{cases} \quad S_v(\Delta z, s_v < 0) \approx \begin{cases} 0, & \Delta z \geq 820 \\ 1, & \text{otherwise} \end{cases} \quad (6.23)$$

This chapter proposes and evaluates two different methods of using these sigmoid functions (a sum and then a product method) to approximate the conditional inequality path constraint. The first method involves summing the horizontal and vertical sigmoid values and the second involves taking the product of these two sigmoids. The following sections detail both methods where for convenience of notation, the exponential terms in equations (6.20) and (6.21) are defined as:

$$\epsilon_h = 1 - \frac{\Delta xy}{2460} \quad (6.24)$$

$$\epsilon_v = 1 - \frac{\Delta z}{820} \quad (6.25)$$

6.4.1 Sigmoid Sum Method.

The sigmoid sum method is the more conservative of the two sigmoid methods. Given the conditional inequality path constraint of satisfying either a horizontal (h) or vertical (v) separation distance constraint, the sigmoid sum approximation of the conditional inequality path constraint appears as:

$$1 - [S_h(\Delta xy, s_h) + S_v(\Delta z, s_v)] \leq 0 \quad (6.26)$$

where s_h and s_v are always positive. Therefore, if $\Delta xy < 2460$ and $\Delta z < 820$, then equation (6.26) is greater than zero. Thus, the sigmoid sum approximation method does

not admit solutions that violate the true conditional inequality path constraint. However, if $\Delta xy > 2460$ while $\Delta z < 820$ or $\Delta xy < 2460$ while $\Delta z > 820$ then equation (6.26) may still be greater than zero, and thus, this method could fail to admit viable solutions that satisfy the true conditional inequality path constraint. The tolerance for when viable solutions are not admitted is a function of the user defined stiffness factor s . The relationships that determines if the sigmoid sum method will admit a viable solution are given by equations (6.27) - (6.29).

Note: Equations (6.27) - (6.44) and the associated text are unique contributions by Maj Arendt that do not appear in [92] and only appear below.

If $\Delta xy \geq 2460$ and $\Delta z \geq 820$ then,

$$\begin{aligned}\epsilon_h &\leq 0 \\ \epsilon_v &\leq 0\end{aligned}\tag{6.27}$$

which implies that $S_h(\Delta xy, s_h) \geq 0.5$ and $S_v(\Delta z, s_v) \geq 0.5$ so equation (6.26) is correctly satisfied.

If $\Delta xy \geq 2460$ and $\Delta z < 820$ then,

$$\begin{aligned}\epsilon_h &\leq 0 \\ \epsilon_v &> 0\end{aligned}\tag{6.28}$$

which implies that $S_h(\Delta xy, s_h) \geq 0.5$ and $S_v(\Delta z, s_v) < 0.5$ so equation (6.26) is correctly satisfied if and only if $\epsilon_v \leq \left| \epsilon_h \right| \frac{s_h}{s_v}$.

If $\Delta xy < 2460$ and $\Delta z \geq 820$ then,

$$\begin{aligned}\epsilon_h &> 0 \\ \epsilon_v &\leq 0\end{aligned}\tag{6.29}$$

which implies that $S_h(\Delta xy, s_h) < 0.5$ and $S_v(\Delta z, s_v) \geq 0.5$ so equation (6.26) is correctly satisfied if and only if $\epsilon_h \leq \left| \epsilon_v \right| \frac{s_v}{s_h}$.

Equations (6.27) - (6.29) guarantee that the sigmoid sum method will only reject feasible solutions if one constraint is violated by more than the slack of the satisfied constraint times the ratio of the two stiffness factors [92]. Furthermore, in the worst-case

when $\Delta z = 0$ then $\epsilon_v = 1$, so the minimum value of Δxy that will satisfy the sigmoid sum approximation of the conditional inequality constraint is given by the following relation:

$$\begin{aligned} &\text{if } \Delta z = 0, \text{ then} \\ &1 - [S_h(\Delta xy, s_h) + S_v(\Delta z, s_v)] \leq 0 \\ &\iff \Delta xy \geq \left(1 + \frac{s_v}{s_h}\right)h \end{aligned} \tag{6.30}$$

Similarly, in the worst-case when $\Delta xy = 0$ then $\epsilon_h = 1$, so the minimum value of Δz that will satisfy the sigmoid sum approximation of the conditional inequality constraint is given by the following relation:

$$\begin{aligned} &\text{if } \Delta xy = 0, \text{ then} \\ &1 - [S_h(\Delta xy, s_h) + S_v(\Delta z, s_v)] \leq 0 \\ &\iff \Delta z \geq \left(1 + \frac{s_h}{s_v}\right)v \end{aligned} \tag{6.31}$$

Equations (6.30) - (6.31) indicate that the worst-case overestimation error can be very large. Additionally, the complexity of the sigmoid sum approximation surface indicates that NLP solvers could have difficulty estimating the gradient of the constraint (see Figure 6.3). Therefore, a second sigmoid function method which reduces much of the overestimation error and improves differentiability is described next.

6.4.2 Sigmoid Product Method.

Compared to the sigmoid sum method, the sigmoid product method allows greater precision in approximating conditional inequality constraints by reducing the maximum overestimation error that occurs when only one constraint is satisfied. Given the conditional inequality path constraint of satisfying a minimum horizontal (h) or vertical (v) separation distance constraint, the sigmoid product approximation of the conditional inequality path constraint appears as [92]:

$$[S_h(\Delta xy, s_h)S_v(\Delta z, s_v)] - 0.25 \leq 0 \tag{6.32}$$

where s_h and s_v are now both negative. The relationships given by equations (6.27) - (6.29) in the sigmoid sum method also determine if the sigmoid product method will admit a

viable solution. However, for the sigmoid product method, if the horizontal constraint is satisfied but the vertical is not, that is, $\Delta xy \geq 2460$ and $\Delta z < 820$ then equation (6.32) is correctly satisfied if:

$$\epsilon_h \leq \frac{1}{s_h} \ln \left(\frac{3 - e^{s_v \epsilon_v}}{1 + e^{s_v \epsilon_v}} \right) \quad (6.33)$$

Similarly, if $\Delta xy < 2460$ and $\Delta z \geq 820$, then equation (6.32) is correctly satisfied if:

$$\epsilon_v \leq \frac{1}{s_v} \ln \left(\frac{3 - e^{s_h \epsilon_h}}{1 + e^{s_h \epsilon_h}} \right) \quad (6.34)$$

In the worst-case when $\Delta z = 0$, then the minimum value of Δxy that will satisfy equation (6.32) is given by the following relationship:

$$\Delta xy \geq \left(1 + \frac{1}{|s_h|} \ln \left(\frac{3 - e^{s_v}}{1 + e^{s_v}} \right) \right) h \quad (6.35)$$

However, $e^{s_v} \rightarrow 0$ as $s_v \rightarrow -\infty$, so equation (6.35) can be approximated conservatively as:

$$\Delta xy \geq \left(1 + \frac{1}{|s_h|} \ln(3) \right) h \quad (6.36)$$

Likewise, when $\Delta xy = 0$, then the minimum value of Δz that will satisfy equation (6.32) is given by the following relationship:

$$\Delta z \geq \left(1 + \frac{1}{|s_v|} \ln \left(\frac{3 - e^{s_h}}{1 + e^{s_h}} \right) \right) v \quad (6.37)$$

which can also be approximated conservatively as:

$$\Delta z \geq \left(1 + \frac{1}{|s_v|} \ln(3) \right) v \quad (6.38)$$

Therefore, the sigmoid product method bounds the overestimation errors (δ_h and δ_v) for infeasible values of Δxy and Δz such that:

$$\begin{aligned} 0 &\leq \delta_h \leq \frac{h}{|s_h|} \ln(3) \\ 0 &\leq \delta_v \leq \frac{v}{|s_v|} \ln(3) \end{aligned} \quad (6.39)$$

From equation (6.39), the maximum overestimation error is strictly a function of $|s_h|$ and $|s_v|$. Due to the computational difficulty resulting from large exponential powers, with 40

fixed collocation nodes the largest absolute values for the sigmoid product parameters that were achieved were $|s_h| = 240$ and $|s_v| = 80$. With these values the maximum overestimation error was 0.5% in the horizontal and 1.4% in the vertical. These values are much lower than the sigmoid sum method while comparable to the maximum overestimation error obtained in the MAES method. As with the MAES method, system designers can select the appropriate value of s_h and s_v based on sensor resolution. The minimum values of $|s_h|$ and $|s_v|$ necessary to guarantee that the overestimation error is less than the sensor tolerance, Δ_s , are given by the following relationships:

$$\begin{aligned} |s_h| &\geq \frac{h}{\Delta_s} \ln(3) \\ |s_v| &\geq \frac{v}{\Delta_s} \ln(3) \end{aligned} \tag{6.40}$$

Since $h = 2460$ and $v = 820$ in the example problem, the minimum value of $|s_h|$ needed to achieve a given tolerance will be $3\times$ greater than the minimum value of $|s_v|$ needed to achieve the same tolerance.

Additionally, Figure 6.3 shows the normalized 3D constraint contour plots for the sigmoid sum and product methods for stiffness values of $s = 4$ and 64 where $s = s_h = s_v$. The blue-colored area in the figure represents the feasible region where at least one constraint is satisfied and the red-colored area represents the infeasible region where neither constraint is satisfied. The thin black line on the constraint surface represents the true feasibility threshold and the thicker dashed line represents the conservative approximation to the feasibility threshold. For the sigmoid sum method the inequality constraint values, equation (6.26), range between ± 1 where negative values indicate at least one constraint is satisfied and values above 0.5 indicate neither constraint is satisfied. In the sigmoid product method the inequality constraint values, equation (6.32), range between -0.25 and 0.75 where negative values indicate at least one constraint is satisfied and values above 0.25 indicate neither constraint is satisfied. Therefore, to compare the two sigmoid methods the constraint contours (\mathbf{g}^*) in Figure 6.3 are normalized using:

$$\mathbf{g}^* = \frac{\mathbf{g} - \mathbf{g}_{\min}}{\mathbf{g}_{\max} - \mathbf{g}_{\min}} \tag{6.41}$$

such that the constraint contour plots for both methods range between 0 and 1 where 0 indicates at least one constraint is satisfied and 1 indicates neither constraint is satisfied.

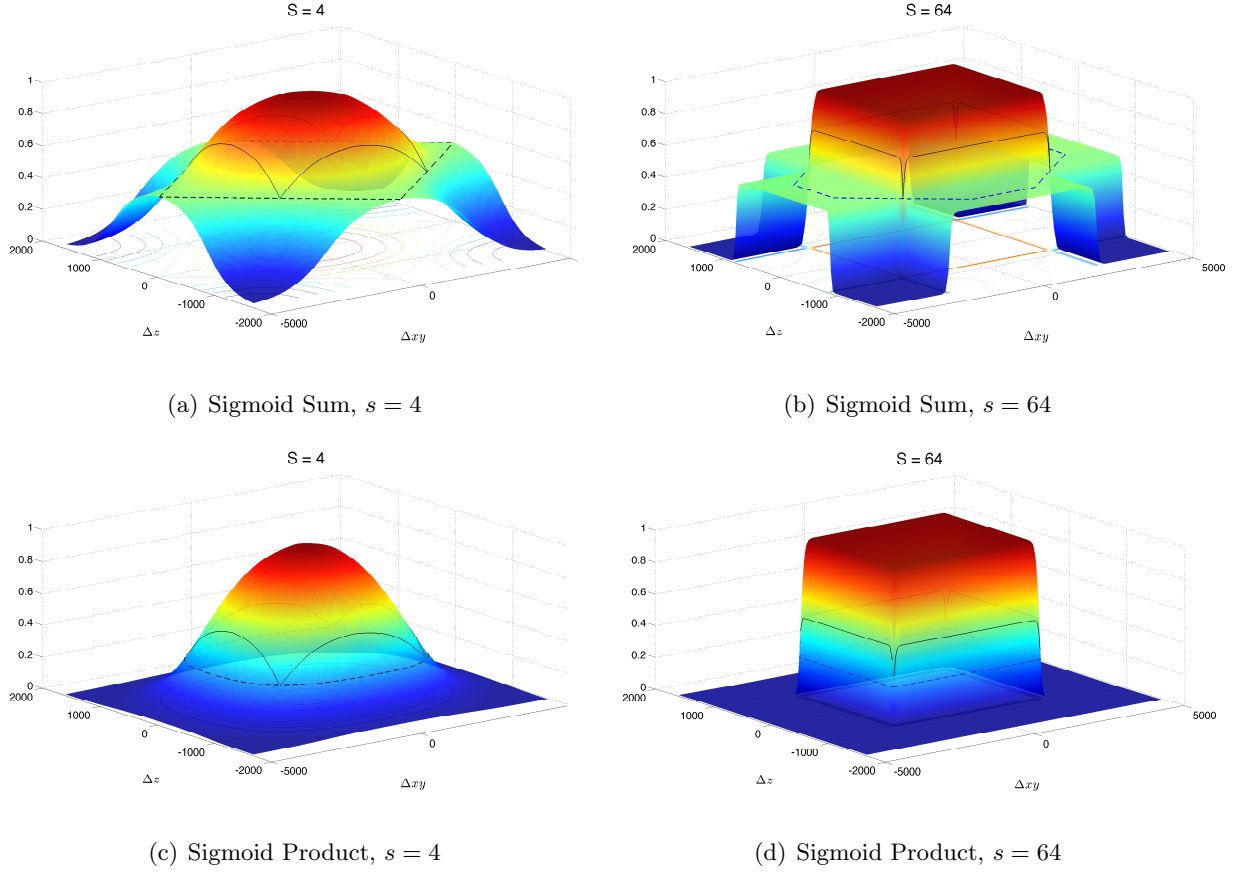


Figure 6.3: 3D Constraint Contour Comparison of Sigmoid Sum and Product Methods

During the optimization, the gradients of these constraint surfaces need to be calculated. Clearly, the sigmoid sum gradient is more complex due to the “stair-steps” and “sharp valleys” in the contour plots and thus is more difficult for the optimizer to establish the correct search direction. Subsequently, as shown in the analysis and confirmed by the results, the sigmoid product method is more efficient and allows higher stiffness values compared to the sigmoid sum method. Thus, the sigmoid product method was selected for use in the general case to resolve optimal control problems with multiple, compound (or *nested*) conditional inequality constraints. For problems of this type, it is necessary to use

the generalized form of the sigmoid product constraint formulation given by [92]:

$$\left(\prod_{k=1}^K \left[1 + e^{s_k \left(\frac{h_k - g_k}{\max\{g_k\} - h_k} \right)} \right]^{-1} \right) - 2^{-K} \leq 0 \quad (6.42)$$

where K is the total number of conditional constraints being evaluated, $g_k \leq \max\{g_k\}$ is a bounded constraint function such that $h_k - g_k \leq 0$ if and only if condition k is satisfied and $h_k - g_k > 0$ if and only if the condition is not satisfied, and $s_k < 0$ is the stiffness factor [92]. The overestimation error for each constraint in the generalized sigmoid product method is bounded similarly to the two-sigmoid case such that [92]:

$$0 \leq \delta_k \leq \left(\frac{\max\{g_k\} - h_k}{|s_k|} \right) \ln(2^K - 1) \quad (6.43)$$

where δ_k is the overestimation error for values that violate conditional constraint k . Equation (6.43) also indicates system designers can select the appropriate value of s_k based on desired precision. The minimum value of $|s_k|$ necessary to guarantee that the overestimation error is less than the precision tolerance, Δ_k , is given by the following relationships [92]:

$$|s_k| \geq \left(\frac{\max\{g_k\} - h_k}{\Delta_k} \right) \ln(2^K - 1) \quad (6.44)$$

Thus, equations (6.39) and (6.40) were used to determine the sigmoid product method parameters for the first example problem, while equations (6.43) and (6.44) were used to determine the sigmoid product method parameters for the second example problem.

6.5 Description of Example Problems

The following section describes the example problems in this chapter. The objective of the first example problem, as described earlier, is for the ownship to minimize deviations from a 3D flight path corridor while maintaining either a horizontal separation distance (Δxy) of at least 2460 ft *or* a vertical separation distance (Δz) of at least 820 ft from an intruder. The objective of the second example problem is identical to the first but requires the ownship to also adhere to FAA right of way (ROW) rules in addition to maintaining the intruder separation distances above. In this problem, the turn direction is conditioned on time and range from the intruder. The setup conditions for both example problems

are identical. For both example problems, the collocation is performed at Legendre-Gauss-Radau quadrature points [101].

6.6 Constraints

Equation (2.3) from Section 2.2.3.2 represents the dynamic constraints for the ownship that the optimization algorithm must satisfy when generating the optimal collision avoidance trajectory. The intruder aircraft maintains a constant speed of 300 ft/sec, a constant heading of 180° and a constant altitude of 6,000 feet. Although the optimal control problem formulation can easily accommodate multiple intruders and more complex and even stochastic models [80], in this example problem we intentionally limited the problem to a single intruder and kept the intruder dynamic constraints simple in order to focus on the methodology for enforcing the conditional inequality path constraints. Thus, the intruder dynamic constraints appear as:

$$\dot{\mathbf{x}}_{\text{int}} = \mathbf{f}(\mathbf{x}_{\text{int}}(t), t) = \begin{bmatrix} \dot{x}_{\text{int}}(t) \\ \dot{y}_{\text{int}}(t) \\ \dot{z}_{\text{int}}(t) \end{bmatrix} = \begin{bmatrix} 300 \cos(180^\circ) \\ 300 \sin(180^\circ) \\ 0 \end{bmatrix} = \begin{bmatrix} -300 \\ 0 \\ 0 \end{bmatrix} \quad (6.45)$$

The equality boundary constraints are time initial (t_0), time final (t_f), ownship initial position (x_0, y_0, z_0), and intruder initial position ($x_{0_{\text{int}}}, y_{0_{\text{int}}}, z_{0_{\text{int}}}$). These boundary constraints appear as:

$$\begin{aligned} t_0 &= 0 \text{ sec} \\ t_f &= 60 \text{ sec} \\ (x_0, y_0, z_0) &= [0, 0, 6000]' \text{ ft} \\ (x_{0_{\text{int}}}, y_{0_{\text{int}}}, z_{0_{\text{int}}}) &= [20000, 2000, 6000]' \text{ ft} \end{aligned} \quad (6.46)$$

The inequality path constraint for the collision avoidance problem is the ownship must maintain at least 2460 feet separation distance horizontally or 820 feet vertically at all time from the intruder. To approximate this conditionally inequality constraint, the MAES, sigmoid sum, and sigmoid product methods are evaluated as described by equations (6.19), (6.26), and (6.32), respectively. In addition, the inequality control constraints, $\mathbf{u}(\mathbf{t})$, appear

as:

$$\begin{aligned} -45^\circ &\leq \mu(t) \leq 45^\circ \\ 0.59 &\leq N_z(t) \leq 1.41 \end{aligned} \tag{6.47}$$

The symmetric control bounds on $N_z(t)$ are reasonable maneuver limits for commercial transport or large remotely piloted aircraft. In addition, the upper bound on $N_z(t)$ corresponds to the upper bound on $\mu(t)$ such that when both controls are at their maximum value the aircraft performs a level turn.

6.7 Performance Measure

The performance measure for this problem is to minimize overall deviation distance (d) from the specified 3D flight path corridor centerline. In Figure 5.8, x_1 and x_2 identify two consecutive waypoints, $x(t)$ the current ownship position, and (d) the deviation distance such that:

$$J = \int_{t_0}^{t_f} d(t) dt \tag{6.48}$$

where the deviation distance, d , is defined by equation (5.36). In this problem, x_1 and x_2 are defined in feet as:

$$\begin{aligned} x_1 &= [0, 2000, 6000]' \\ x_2 &= [25000, 2000, 6000]' \end{aligned} \tag{6.49}$$

Note that the ownship is trying to fly along the line through x_1 and x_2 , but there is no time specified with either point.

6.8 Example Problem 1

In this section we first analyze the results of using the MAES to approximate the inequality path constraint and then examine the results of using the sigmoid methods for the first example problem. For comparison, in each case we used a global polynomial with 40 fixed collocation nodes and used IPOPT as the NLP solver. Simulations in this chapter used Matlab[®] version 2012b on a laptop computer operating with OS X version 10.9 operating system and a 2.3 GHz Intel Core *i5* processor with 16 GB 1333 MHz DDR3 memory. The performance measure for this scenario was to minimize path deviation, as

in equation (6.48). Since the minimum horizontal separation distance was $3\times$ greater than the minimum vertical separation distance, intuitively the minimum deviation trajectory was for the ownship to intercept the 3D flight path corridor and change altitude only when required to meet the conditional separation constraint. The simulation results confirmed this intuition. The primary differences in these two approaches was the accuracy of the approximation and the time required for the optimizer to achieve a solution.

To standardize the results we provided the NLP solver with the same conservative initial guess for each simulation run which consisted of the ownship flying level at the initial condition heading of 0 degrees. The minimum path deviation trajectory for all three methods appeared similar. Although Figure 6.4 is a time-series quad chart of only the MAES simulation results, the minimum path deviation trajectories from the sigmoid methods appeared the same as the results in this figure. In this figure, the ownship is depicted in blue, the intruder in red, and the desired 3D flight path corridor in black. Both aircraft started the scenario at the same altitude of 6,000 feet MSL with the ownship 2,000 feet south (positive y-axis) of the 3D corridor while the intruder started and remained on the corridor.

As seen in panel (a) of Figure 6.4, at the start of the scenario the ownship began an immediate left turn to intercept the 3D flight path corridor and then continued on the corridor at the specified corridor altitude of 6,000 feet as shown in panel (b). In panel (c), the ownship then deviated from the corridor altitude by climbing only when required to meet the conditional separation inequality constraint. After satisfying this conditional constraint, the ownship then descended and maintained the desired 3D flight path corridor for the duration of the time horizon shown in panel (d).

6.9 MAES Simulation Results

Table 6.1 summarizes the effects of increasing the exponential terms in equation (6.19) on the normalized cost (J), number of NLP inequality constraint evaluations, CPU time in NLP evaluations, and the vertical and horizontal separation distances from the intruder at the closest point of approach (CPA).

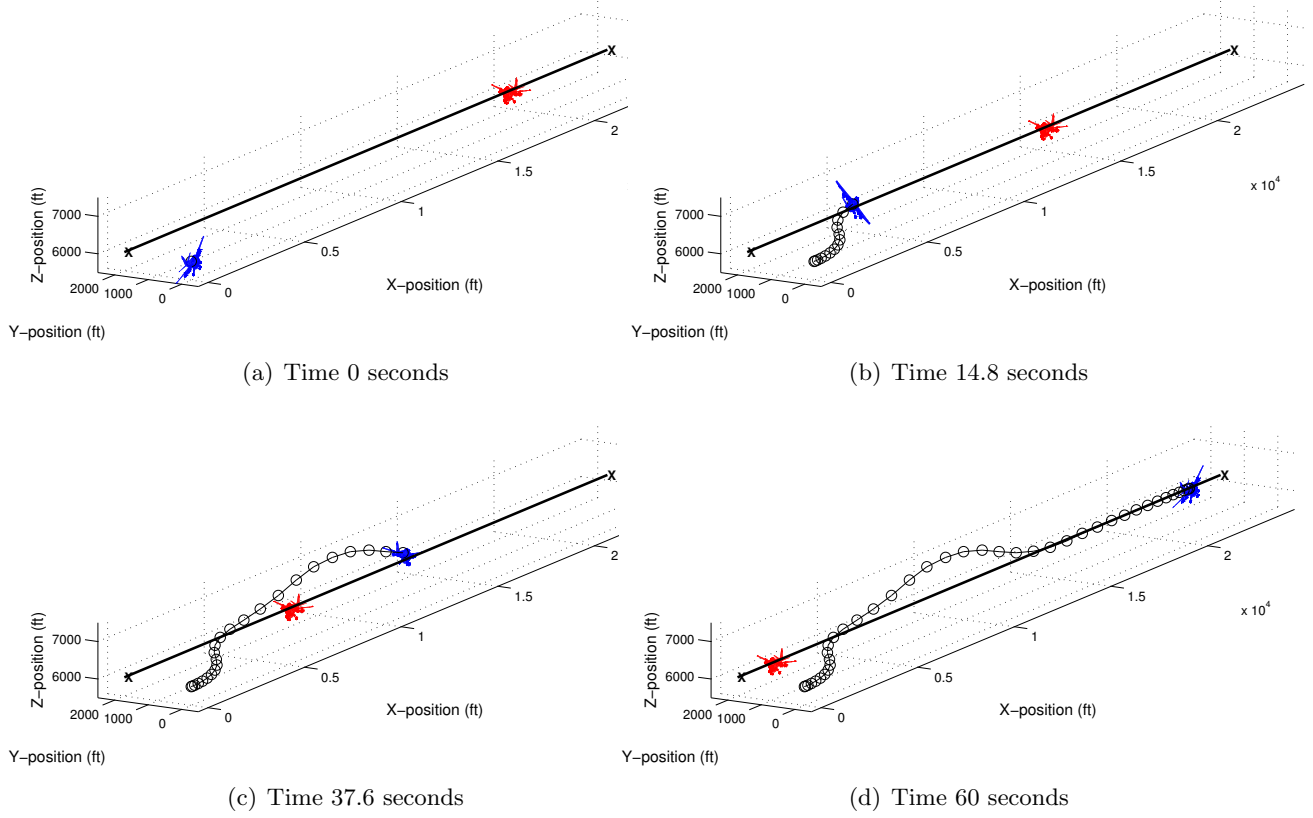


Figure 6.4: Time Series of Optimal Trajectory for Ownship (Blue) Avoiding the Intruder Aircraft (Red) While Minimizing Path Deviation (MAES, Sigmoid Results Similar).

Table 6.1: Results for MAES Approximation of Conditional Inequality Constraint

MAES Order (N)	Normalized Cost (J)	# of Inequality Constraint Evaluations	CPU time in NLP Evaluations (sec)	Separation at CPA	
				Vertical (ft)	Horizontal (ft)
2	1.666	130	26.41	1129	795
4	1.46	59	23.32	974	775
100	1.344	64	22.8	875	772
200	1.341	69	27.26	872	761

The normalized cost (J) in Table 6.1 represents the ratio of the cost of intercepting the 3D corridor with an avoidance maneuver normalized by the cost of intercepting the 3D corridor without an avoidance maneuver. Note that these results are for 40 collocation nodes. While the number of nodes will affect the results, increasing the number of nodes will not necessarily cause the generated trajectory at the CPA to achieve the minimum feasible

separation distances. In fact, due to the interaction between the aircraft dynamics and cost function, equations (2.3) and (6.48), the minimum vertical and horizontal separation distances at the CPA may overshoot the minimum feasible separation distances of the active constraint for any number of collocation nodes. Figure 6.5 graphically displays the

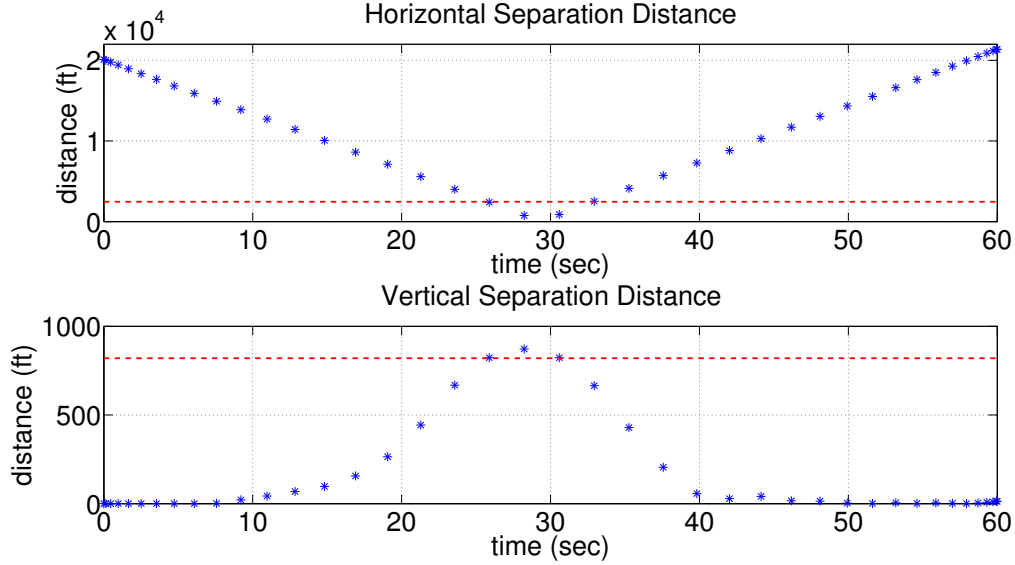


Figure 6.5: Results Using $N = 200$ for MAES Approximation of Inequality Path Constraint

results for $N = 200$. The blue asterisks in the plots show the horizontal (Δxy) and vertical (Δz) separation distances between the ownship and the intruder aircraft respectively at each collocation node for the 60 second time-horizon. The red-line in each plot depicts the minimum horizontal (2460 ft) or vertical (820 ft) separation distance. Figure 6.5 shows that at approximately 12 seconds, the ownship began a climb so that as the horizontal separation decreased to below 2460 ft, at approximately 26 seconds, the ownship achieved the required vertical separation of at least 820 ft. In this plot, the ownship climbed above the minimum altitude of 820 ft and peaked at an altitude of approximately 870 ft. Although it appears the separation distance was slightly violated past 30 seconds, this is because the solution is satisfied at the collocation nodes only. The results for both sigmoid methods appeared similar to the results in Figure 6.5.

6.10 Sigmoid Simulation Results

6.10.1 Sigmoid Sum Results.

Table 6.2 summarizes the results of increasing the stiffness factors (s_h and s_v) in equation (6.26) on the cost (J), number of inequality constraint evaluations, CPU time in NLP evaluations, maximum vertical separation distance from the intruder, and the minimum horizontal separation distance from the intruder.

Table 6.2: Results for Sigmoid Sum Approximation of Conditional Inequality Constraint

Stiffness factor (s_h, s_v)	Normalized Cost (J)	# of Inequality Constraint Evaluations	CPU time in NLP Evaluations (sec)	Separation at CPA	
				Vertical (ft)	Horizontal (ft)
(50, 50)	1.656	583	228.70	1122	790
(100, 100)	1.458	1174	365.56	971	776
(125, 125)	1.423	1189	400.82	941	777

6.10.2 Sigmoid Product Results.

Table 6.3 summarizes the results for the sigmoid product method of increasing the stiffness factors (s_h and s_v) in equation (6.32) on the cost (J), number of inequality constraint evaluations, CPU time in NLP evaluations, maximum vertical separation distance from the intruder, and the minimum horizontal separation distance from the intruder.

Table 6.3: Results for Sigmoid Product Approximation of Conditional Inequality Constraint

Stiffness factor (s_h, s_v)	Normalized Cost (J)	# of Inequality Constraint Evaluations	CPU time in NLP Evaluations (sec)	Separation at CPA	
				Vertical (ft)	Horizontal (ft)
(180, 60)	1.443	121	50.81	825	989
(210, 70)	1.422	128	52.9	824	968
(240, 80)	1.403	80	38.69	824	915

6.11 Sensor Tolerance Evaluation Results

The sigmoid sum method had the largest computational time and was the most conservative of the three methods. Therefore, the following sensor tolerance evaluation focuses on the performance of the MAES and sigmoid product methods with parameters chosen to guarantee maximum overestimation errors less than a given sensor tolerance. For simplicity, this evaluation assumes the horizontal and vertical sensor tolerances are equal. From the previous example, for a sensor tolerance of ± 12 feet, the minimum value of N for

the MAES method to guarantee the maximum overestimation error is less than the sensor tolerance is $N = 144$. Similarly, from equation (6.40) for the sigmoid product method the minimum values of $|s_h|$ and $|s_v|$ required to guarantee the maximum overestimation error is less than the sensor tolerance is $|s_h| = 226$ and $|s_v| = 76$. Table 6.4 shows the results for the MAES and sigmoid product methods with parameter values that guarantee the maximum overestimation error is less than sensor tolerances of 12, 25 and 50 feet.

The previous results in Sections 6.9 and 6.10 used only 40 collocation nodes and these nodes spanned the entire trajectory as a single “global” interpolating polynomial. However, to increase the fidelity of the solution especially near the constraint activation boundaries, the results in Table 6.4 divided the trajectory into 20 equal-spaced segments with 10 collocation nodes per segment [63]. Although not reflected in the table, an alternate formulation applied an adaptive mesh refinement strategy [101] which adaptively increased the number and placement of collocation nodes to achieve a user-defined level of accuracy. However, since the execution times for the adaptive node placement strategy varied significantly based on the number of mesh refinements, for standardization and comparison of results, a fixed number of collocation nodes was preferred for this analysis. Furthermore, due to the longer execution times of an adaptive node placement strategy, any eventual implementation of a real-time airborne collision avoidance algorithm would likely use fixed collocation nodes.

Table 6.4: Comparison of Methods for Achieving Error Less Than Sensor Tolerance

Sensor Tolerance (ft)	Method	Normalized Cost (J)	Constraint Activation times (sec)	CPU time in NLP (sec)	Separation at CPA	
					Vertical (ft)	Horizontal (ft)
12	MAES $N = 144$	1.399	$t_1 = 25.69$ $t_2 = 32.73$	254.88	919	116
	Sigmoid Product $s_h = -226$ $s_v = -76$	1.394	$t_1 = 25.81$ $t_2 = 32.86$	346.13	929	102
25	MAES $N = 70$	1.406	$t_1 = 25.81$ $t_2 = 32.86$	236.54	918	110
	Sigmoid Product $s_h = -109$ $s_v = -37$	1.396	$t_1 = 25.81$ $t_2 = 32.85$	285.51	931	102
50	MAES $N = 36$	1.401	$t_1 = 25.80$ $t_2 = 32.85$	147.1	936	107
	Sigmoid Product $s_h = -55$ $s_v = -19$	1.399	$t_1 = 25.81$ $t_2 = 32.86$	162.75	934	107

To better gauge the changes in the ownship trajectory as a function of the change in the sensor tolerance, the results in Table 6.4 replaced the column showing the number of inequality constraints evaluations in Tables 6.1 - 6.3 with a new column that showed the constraint activation times. As seen in Figure 6.5, the intersection of the red-dashed line and blue-asterisk indicate the constraint activation times. For the sensor tolerance evaluation, changes in the constraint activation times can provide additional insight into the sensitivity of the aircraft dynamics to the sensor tolerance. For instance, the performance measure (J) in this problem should force the optimal trajectory towards the “corner” of the constraint boundary where the horizontal and vertical constraints are active since in both MAES and sigmoid methods, these constraint corners are where the overestimation error is zero. This fact is particularly evident in Figure 6.1 where the formulation of the MAES optimization problem in equation (6.7) minimized the overestimation error at the corners of the rectangular constraint area.

In general the normalized cost (J) in Table 6.4 increased slightly as the predicted overestimation error increased. However, the constraint activation times remained consistent with the constant ground speed assumption, and the minimum separation distances at the CPA did not noticeably change as the sensor tolerance increased. These results indicate that the aircraft dynamics and trajectory optimization process were not sensitive to the range of sensor tolerances in the table. Since the overestimation error achieved its minimum value at the constraint corner, the optimizer forced the trajectory to intersect this corner as seen by the consistent constraint activation times. Even at a sensor tolerance of 50 feet, the ownship dynamic constraints were still what drove the optimal trajectory to start a climb away from the desired flight path in order to intersect the constraint corner; that is, the ownship climbed to reach an altitude of 820 feet above the intruder at the exact moment the horizontal separation distance decreased to less than 2460 feet. Likewise, after the two aircraft passed, the ownship then descended below 820 feet above the intruder at the exact moment when the horizontal separation distance again increased to greater than 2460. Thus, the trajectory was insensitive to sensor tolerances up to 50 feet. This was because the ownship's dynamics forced the aircraft to climb to intersect the constraint corner rather than to avoid the worst-case overestimation region corresponding to sensor tolerances up to 50 feet. As a result, based on the intercept geometry of this example problem, the optimal trajectory should not change significantly until the sensor tolerance is significantly greater than the aircraft's dynamic constraints required to intersect the constraint corner. For example, the earlier MAES results with $N = 2$ correspond to sensor tolerances of greater than 330 feet which was reflected in the fact that at the CPA the altitude separation was approximately 360 feet greater than the minimum required separation distance. Therefore, based on the intercept geometry the aircraft dynamics may be more important in determining the precision of the approximation rather than the sensor tolerance since the optimal trajectory may remain unchanged for varying values of realistic sensor tolerances and scaling may not be required.

Nonetheless, Table 6.4 confirmed the methods presented in the chapter and provides users a means to implement conditional inequality path constraints with a gradient-based numerical solver to the desired level of precision. Additionally, the results confirm that if the problem involves only two simple constraints, then the MAES method is the superior approximation method.

6.12 Example Problem 2

Unlike the previous example problem of satisfying a minimum horizontal or vertical separation distance where the MAES method performed well, an optimal control problem formulation may include multiple, compound (or *nested*) conditional constraints that do not lend themselves practically to the MAES formulation. An example of this type of complication is adhering to FAA right of way (ROW) rules, which state that if two aircraft are approaching nearly head on, then “each aircraft shall alter course to the right.” Since air traffic control procedures prefer horizontal over vertical maneuvers to maintain safe separation, in addition to implementing this conditional ROW constraint, this example problem also uses a new weighted cost function that separately penalizes ownship horizontal and vertical deviations from a desired 3D flight path corridor. This new cost function appears as,

$$J = \int_{t_0}^{t_f} \left[\left(\frac{d_{xy}(t)}{3038} \right)^2 + \left(\frac{d_z(t)}{300} \right)^2 \right] dt \quad (6.50)$$

where $d_{xy}(t)$ is the horizontal deviation from the 3D corridor centerline and $d_z(t)$ is the vertical deviation. The quadratic penalty in equation (6.50) is based on an assumed 3D corridor defined as ± 3038 feet (half a nautical mile) horizontally and ± 300 feet vertically from centerline. Because this new cost function will likely cause the ownship to maneuver horizontally instead of vertically to maintain safe separation, the ownship will need to comply with the horizontal ROW constraint and alter course to the right since their approach will be nearly head-on in this example problem.

6.13 Right of Way Formulation

In formulating the conditional ROW constraint, the sigmoid product method is used to implement a set of conditional logic *'if statements.'* The feasibility region for the conditional ROW constraint is described by the following set of compounded logic *OR* conditions: If the separation distance between the ownship and intruder is greater than or equal to $2 \times$ the horizontal keep out radius of 2460 feet *OR* the time to CPA (T_{CPA}) is greater than or equal to 30 seconds *OR* time from CPA is less than or equal to -5 seconds *OR* the relative azimuth angle (θ) between the ownship and the intruder is greater than or equal to zero (so the ownship will pass to the right of the intruder) then the solution is feasible; otherwise, the trajectory is not feasible. This inequality constraint formulation appears algorithmically as:

```

if  $\left( \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \geq 2 \times 2460 \text{ feet} \right)$ 
    feasible
else if  $(T_{\text{CPA}} \leq -5 \text{ seconds})$ 
    feasible
else if  $(T_{\text{CPA}} \geq 30 \text{ seconds})$ 
    feasible
else if  $(\theta \geq 0)$ 
    feasible
else
    infeasible
end

```

Each of the four conditional constraints in the ROW formulation are approximated using unique sigmoid functions. The range separation indicator function approximation at each point appears as:

$$S_r(\Delta x, \Delta y, \Delta z, s_r) = \left[1 + e^{-s_r \left(1 - \frac{\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}}{2 \times 2460} \right)} \right]^{-1} \quad (6.51)$$

Thus, when $\left(\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} < 2 \times 2460\right)$, the range indicator function approximation is active. By assuming a constant velocity and solving for the time that minimizes the instantaneous separation distance, the time to CPA (T_{CPA}) appears as:

$$T_{\text{CPA}} = - \frac{[\Delta x \ \Delta y \ \Delta z] [\Delta v_x \ \Delta v_y \ \Delta v_z]^T}{[\Delta v_x^2 + \Delta v_y^2 + \Delta v_z^2]} \quad (6.52)$$

where negative values indicate the two aircraft have passed or their velocity vectors are on non-convergent paths. Thus, the T_{CPA} indicator function approximations appear as:

$$\begin{aligned} S_{t_{\text{entry}}}(T_{\text{CPA}}, s_t) &= \left[1 + e^{s_t \left(\frac{T_{\text{CPA}} - T_{\text{high}}}{T_{\text{high}} - T_{\text{low}}} \right)} \right]^{-1} \\ S_{t_{\text{exit}}}(T_{\text{CPA}}, -s_t) &= \left[1 + e^{s_t \left(\frac{T_{\text{CPA}} - T_{\text{low}}}{T_{\text{high}} - T_{\text{low}}} \right)} \right]^{-1} \end{aligned} \quad (6.53)$$

where $T_{\text{low}} = -5$ seconds and $T_{\text{high}} = 30$ seconds, and when $(T_{\text{CPA}} > -5)$ OR $(T_{\text{CPA}} < 30)$ the T_{CPA} indicator function approximation is active. Finally, the turn direction constraint (S_θ) is formulated based on relative azimuth angle (θ) where,

$$\theta = \tan^{-1} \left[\frac{\Delta y}{\Delta x} \right] \quad (6.54)$$

and is approximated at each instance in time using the following sigmoid function,

$$S_\theta(\theta, s_\theta) = \left[1 + e^{s_\theta \left(1 - \frac{\tilde{\theta}}{\pi} \right)} \right]^{-1} \quad (6.55)$$

where $\tilde{\theta} = \theta + \pi$. Thus, when $(\theta > 0)$ the “right turn” constraint approximated by equation (6.55) is satisfied. Therefore, based on equation (6.42) with $K = 4$, the approximation of the ROW conditional inequality path constraint appears as [92]:

$$[S_r(\Delta x, \Delta y, \Delta z, s_r) S_{t_{\text{entry}}}(T_{\text{CPA}}, s_t) S_{t_{\text{exit}}}(T_{\text{CPA}}, -s_t) S_\theta(\theta, s_\theta)] - 0.0625 \leq 0 \quad (6.56)$$

where $S_r(\Delta x, \Delta y, \Delta z, s_r)$, $S_{t_{\text{entry}}}(T_{\text{CPA}}, s_t)$, $S_{t_{\text{exit}}}(T_{\text{CPA}}, -s_t)$ and $S_\theta(\theta, s_\theta)$ are defined in equations (6.51), (6.53), and (6.55), respectively.

6.13.1 Simulation Results and Analysis.

As described in Section 6.5, the setup for this second example problem is identical to the first problem; however, the cost function in equation (6.48) is now replaced by the weighted cost function in equation (6.50).

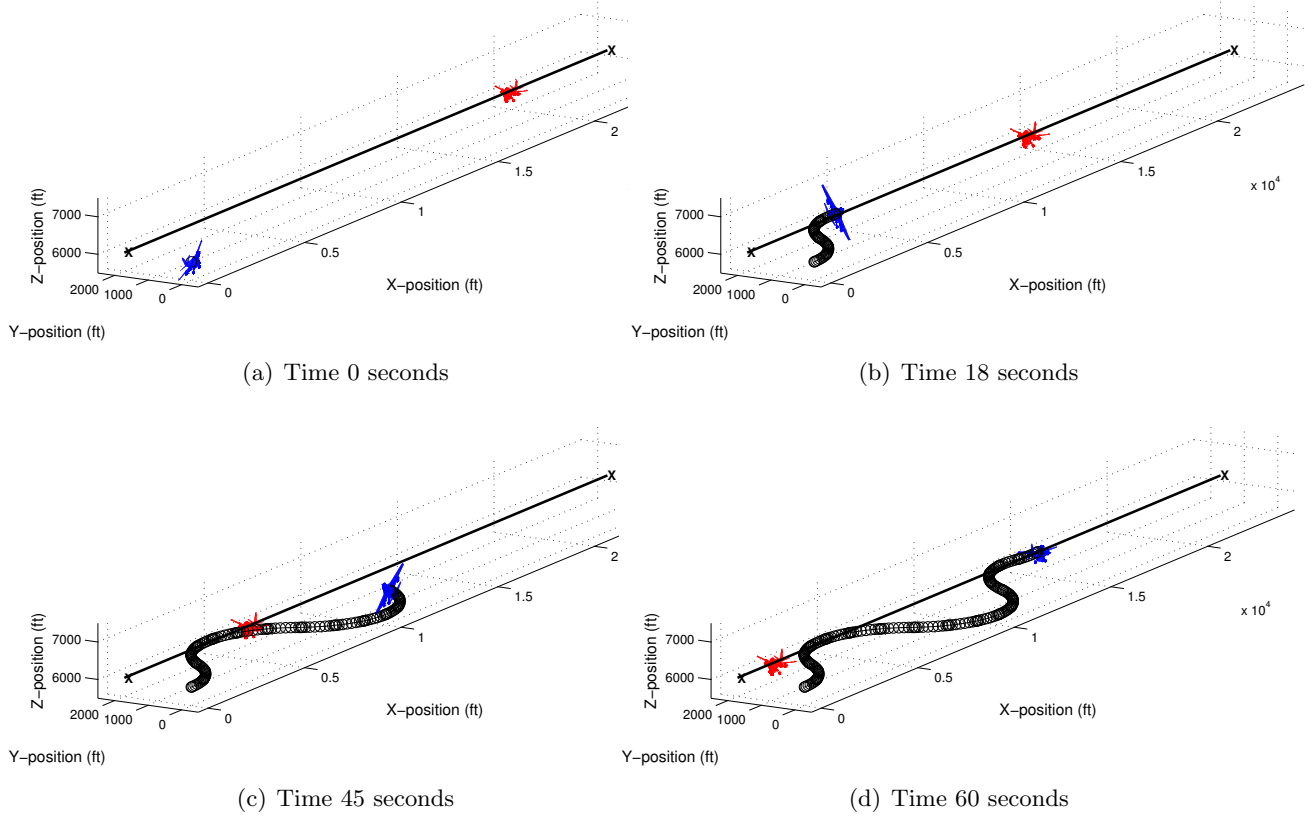


Figure 6.6: Time Series of Optimal Trajectory for Ownship (Blue) Avoiding the Intruder Aircraft (Red) by Adhering to Right of Way While Minimizing Path Deviation.

In addition, the ownship must now not only satisfy the conditional inequality path constraint in equation (6.19) formulated using the MAES method ($N = 200$), but also satisfy the conditional inequality ROW constraint in equation (6.56) formulated using the sigmoid product method ($s_t = s_r = s_\theta = 200$). Like the sensor tolerance evaluation in the first example problem, this example divided the trajectory into 20 equal-spaced segments with 10 collocation nodes per segment. Figure 6.6 shows the simulation results. As in the first example problem, at the start of the scenario the ownship immediately maneuvered

north (positive y axis) to minimize the path deviation from the 3D flight path corridor. However, due to the weighted cost function the ownship now maneuvered horizontally instead of vertically to keep out of the minimum separation distance from the intruder and correctly altered course to the right to comply with the conditional horizontal ROW constraint.

This example problem demonstrated that the sigmoid product method can effectively resolve multiple conditional constraints, to include constraints that are not naturally bounded (such as conditions that involve time or variables that are unrestricted in sign), and offers a robust alternative for problems where the MAES method is not suitable. Further, even with four conditional constraints as in this example problem, the error bounds for the sigmoid product method are valid. For example, based on equation (6.43) with $s_t = s_r = 200$, the maximum overestimation error for the time and range conditional constraints was only 2.7%. Nevertheless, the use of this approach requires an understanding of the potential limitations. For instance, a well-known and often-stated limitation of gradient-based NLP search methods is they produce local optimal solutions, which may or may not be global solutions. The formulation and testing of the ROW formulation highlighted the potential applicability of this limitation in the context of airborne collision avoidance. For instance, given identical initial conditions, to enforce a “left turn” constraint required an initial trajectory guess to the left in order for the optimizer to locate the global vice the local optimal solution. A follow-on research effort explores potential methods such as those listed in [35] for appropriately choosing “smart” initial guesses for complex compounded conditional constraints. Another important consideration is the number of collocation nodes and stiffness of the sigmoid function. For instance, if the nodes are too sparse then the sigmoid appears as a binary switching function causing the NLP to fail since the conditional constraint approximation is no longer differentiable. Thereby, the number and location of collocation nodes along with the sigmoid stiffness plays an important role in determining differentiability of the conditional constraint. Besides increasing collocation nodes and/or decreasing the sigmoid stiffness factor, an additional remedy to this situation

is to use an adaptive mesh refinement strategy [101] as previously discussed which adaptively increases the number and placement of collocation nodes to help maintain differentiability of the conditional constraints approximated by the sigmoid functions. A final consideration when using this method is the potential for long convergence times. With 200 collocation nodes the NLP took 184.5 seconds to converge to a solution; however, in this chapter the simulation algorithms were not necessarily optimized for speed but were coded for robust post-processing analysis. For real-time implementation these convergence times will need to be improved using techniques such as parallel processing or a more efficient programming language.

6.14 Conclusion

This chapter motivated the application of conditional inequality path constraints in the nonlinear airborne collision avoidance optimal control problem. This chapter then developed and demonstrated two different methods to enforce conditional inequality path constraints using numerical gradient-based solvers by approximating the mixed-norm and indicator function classes of constraint formulations. In addition, this chapter analytically derived the maximum overestimation error bounds associated with these different approximation methods and also provided designers a means to determine the minimum computational complexity needed to achieve desired results based on sensor performance. Using realistic collision avoidance scenarios, this chapter demonstrated the performance of these methods and confirmed the validity of the error bounds. Furthermore, both the minimum area enclosing superellipse (MAES) and sigmoid product methods yielded good results; however, due to the geometric intuition and faster computation times the MAES method may be more advantageous for normalized and non-complex constraints. However, the MAES method is not well-suited if the conditional constraints are not continuous or if the constraints are compounded. In these cases, the sigmoid product method provides a robust means to satisfy conditional constraints and has good error bounds. Having developed all the necessary ‘pieces’, the next step is to compare the optimal control approach developed

herein to the existing Jointly Optimal Collision Avoidance (JOCA) algorithm used in the Multiple Intruder Autonomous Avoidance (MIAA) program.

VII. Comparison of Trajectory Optimization Methods

THE FOLLOWING chapter is a limited comparison between the Jointly Optimal Collision Avoidance (JOCA) algorithm used in the Multiple Intruder Autonomous Avoidance (MIAA) program [2] and the optimal control algorithm developed through the work herein. The motivation for this comparison is to address the question “How do the trajectories of an optimal control approach compare to the JOCA algorithm results when generating an airborne collision avoidance solution?” In order to focus on the differences of the algorithms, the scope of this comparison is intentionally limited to non-stochastic inputs. Thus, both algorithms will use the same deterministic inputs for the intruder.

The ownship-intruder setup geometries for this evaluation are common collision avoidance test geometries for the MIAA program and similar to the scenarios flown during the airborne flight test program where the JOCA algorithm was flown onboard a manned surrogate platform acting as the RPA. The objective or cost functional for these scenarios is to minimize deviation from a 3D flightpath (corridor centerline) over a finite time interval $t_f - t_0$, normalized to the corridor size where the corridor is defined as ± 3038 feet (or half a nautical mile) of centerline in the horizontal (Δh) direction and ± 600 feet of centerline in the vertical (Δv) direction. This cost functional appears as,

$$J = \int_{t_0}^{t_f} \left[\left(\frac{\Delta h(t)}{3038} \right)^2 + \left(\frac{\Delta v(t)}{600} \right)^2 \right] dt \quad (7.1)$$

The measurements in these simulations consist of simulated 1 Hz ADS-B position and velocity measurements for each intruder. These measurements are assumed not noise corrupted and represent the intruders’ true position and velocity. In these scenarios the intruders are non-maneuvering. The inequality path constraint as defined in Section 6.2 is that the ownship must maintain at least 2460 feet separation distance horizontally or 820 feet vertically from all intruders. To approximate this conditional inequality constraint, the optimal control algorithm developed in the work herein applies the MAES method from

Chapter VI as described by equation (6.19) and repeated here for convenience appears as,

$$\ln 2 - \ln \left(\left(\frac{\Delta xy}{2460} \right)^N + \left(\frac{\Delta z}{820} \right)^N \right) \leq 0 \quad (7.2)$$

where N is an even natural number larger than two.

7.1 Model Parameters

In order to objectively compare the trajectories between the two algorithms, the ownship dynamics model must match the JOCA model so it must now account for actuator dynamics as well as rate limits. The ownship dynamic constraints [2] from Section 2.2.3.2, which are repeated here for convenience in equation (7.3), now need to be appended with the actuator dynamics in equation (7.4).

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{z}(t) \\ \dot{\gamma}(t) \\ \dot{\chi}(t) \end{bmatrix} = \begin{bmatrix} V \cos \gamma(t) \cos \chi(t) \\ V \cos \gamma(t) \sin \chi(t) \\ V \sin \gamma(t) \\ \frac{N_z g \cos \mu(t) - g \cos \gamma(t)}{V} \\ \frac{N_z g \sin \mu(t)}{V \cos \gamma(t)} \end{bmatrix} \quad (7.3)$$

Based on this dynamics model, the optimization algorithm calculates the desired optimal control time histories, where N_{z_c} and μ_c commands are then sent to the flight control system where the inner-loop controller moves the appropriate flight control surface(s) to achieve the desired N_z and μ . As a result of the system dynamics the aircraft does not achieve the commanded normal acceleration and bank angle instantaneously. These system delays are both approximated using second-order dynamic models with the following transfer functions [2]:

$$\begin{aligned} \frac{N_z}{N_{z_c}} &= \frac{\omega_{n_z}^2}{s^2 + 2\zeta_z \omega_{n_z} s + \omega_{n_z}^2} \\ \frac{\mu}{\mu_c} &= \frac{\omega_{n_\mu}^2}{s^2 + 2\zeta_\mu \omega_{n_\mu} s + \omega_{n_\mu}^2} \end{aligned} \quad (7.4)$$

where ω_{n_z} and ω_{n_μ} are the natural frequency and ζ_z and ζ_μ the damping ratio for N_z and μ , respectively. The values for these coefficients in equation (7.4) are supplied by the research sponsor and are based on the notional performance of a Group 5 UAS and are the

same values used in the JOCA algorithm during this comparison evaluation. Combining equations (7.3) and (7.4), the ownship dynamics constraint equation now appears as,

$$\dot{x} = \mathbf{f}(x(t), \mathbf{u}(t), t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{z}(t) \\ \dot{\gamma}(t) \\ \dot{\chi}(t) \\ \dot{N}_z(t) \\ \dot{N}_{z2}(t) \\ \dot{\mu}(t) \\ \dot{\mu}_2(t) \end{bmatrix} = \begin{bmatrix} V \cos \gamma(t) \cos \chi(t) \\ V \cos \gamma(t) \sin \chi(t) \\ V \sin \gamma(t) \\ \frac{N_z g \cos \mu(t) - g \cos \gamma(t)}{V} \\ \frac{N_z g \sin \mu(t)}{V \cos \gamma(t)} \\ N_{z2} \\ -2\zeta_z \omega_{n_z} N_{z2} - \omega_{n_z}^2 (N_z - N_{zc}) \\ \mu_2 \\ -2\zeta_\mu \omega_{n_\mu} \mu_2 - \omega_{n_\mu}^2 (\mu - \mu_c) \end{bmatrix} \quad (7.5)$$

where N_z and μ are now included as dynamic constraints with N_{z2} and μ_2 appended as additional states to complete the model of the second-order dynamics in equation (7.4). Further, based on the performance characteristic of the notional Group 5 UAS, the upper and lower control bounds appear as:

$$\begin{aligned} 0.75 \leq N_{zc}(t) \leq 1.25 \\ -15^\circ \leq \mu_c(t) \leq 15^\circ \end{aligned} \quad (7.6)$$

In addition, the rate-limit for the bank angle (μ) command is specified explicitly in the JOCA model as ± 8 deg/sec for the notional Group 5 UAS platform; however, the rate-limit for the normal acceleration (N_z) command is not explicitly specified. Therefore, the optimal control algorithm developed for the work herein applied a rate-limit of ± 0.25 g/sec; nevertheless, based on the coefficients in equation (7.4) this imposed rate-limit is inconsequential since the N_z rate response is much lower than this limit and is consistent with the magnitude of the N_z rate response in the corresponding JOCA solutions. The control rate-limits appear as:

$$\begin{aligned} -0.25 \text{ g/sec} \leq \dot{N}_{zc}(t) \leq 0.25 \text{ g/sec} \\ -8 \text{ deg/sec} \leq \dot{\mu}_c(t) \leq 8 \text{ deg/sec} \end{aligned} \quad (7.7)$$

In order to enforce the control rate-limits in equation (7.7), the optimal control algorithm applies the rate-limit bounds on the state derivatives, N_{z2} and μ_2 , of N_z and μ in equation (7.5). Since the “filtered” states N_{z2} and μ_2 are not directly the commanded control rate states, to ensure the commanded control rate limits are not violated this research applies a conservative bound on N_{z2} and μ_2 of 90% of the appropriate control rate-limit in equation (7.7). Applying this conservative bound on the filtered control-rate states eliminates the need to add any additional states to the dynamics model to capture this behavior. Thus, the bounds on the filtered control rate states appear as,

$$\begin{aligned} -0.225 \text{ g/sec} &\leq \dot{N}_{z2}(t) \leq 0.225 \text{ g/sec} \\ -7.2 \text{ deg/sec} &\leq \dot{\mu}_2(t) \leq 7.2 \text{ deg/sec} \end{aligned} \tag{7.8}$$

With these conservative bounds in equation (7.8), the optimal control solution did not exceed the commanded control limits in equation (7.7) for the evaluations in this chapter. Finally, since the bounds of the flightpath angle (γ) are not explicitly specified in the JOCA model, based on the performance characteristics of the notional Group 5 UAS the optimal control algorithm applies a conservative bound of $\pm 11^\circ$ for the flightpath angle, γ .

7.2 Algorithm Description

For this comparison both the JOCA algorithm and the optimal control algorithm developed for this research use a receding horizon implementation with a fixed time horizon of 30 seconds along with a 1 Hz update rate. A longer time horizon or faster update rate would improve collision avoidance performance, yet at a cost of increased computational processing. In light of typical flight operations in the National Airspace System (NAS), the 1 Hz rate is used which is consistent with and mirrors the current TCAS 1 Hz update rate [2].

7.2.1 JOCA Description.

The JOCA algorithm generates a predetermined set of 30-second candidate avoidance trajectories based on the RPA’s mission waypoints [2]. To allow for greater diversity of candidate avoidance maneuvers these 30-second trajectories are “stitched” together by two

15-second trajectories. The candidate avoidance trajectories in each 15-second trajectory are generated using fixed increments of N_z and μ commands [2]. For this evaluation the JOCA algorithm uses 10 distinct combinations of N_z and μ commands per 15-second stitch. This combination of 2 stitches and 10 distance maneuver commands per 15-second stitch results in a total of 100 candidate trajectories per 30-second time horizon [2].

7.2.2 Optimal Control Description.

The optimal control algorithm plans and calculates the optimal set of control inputs over a fixed 30-second time interval, flies the first time step while planning and calculating again the next 30-second interval. This pattern is repeated which effectively transforms the optimization algorithm “from a static planner into a dynamic planner” [73]. In the optimal control approach, the previous control solution is the initial guess for the subsequent planning horizon. The optimization algorithm for this analysis is IPOPT.

When planning an avoidance solution, both the optimal control and JOCA algorithms account for all intruders in the time horizon by simultaneously calculating a concurrent solution vice pairwise solutions. However, unlike the JOCA algorithm which uses a compiled language and operates in near real-time, the current implementation of the optimal control method uses Matlab[®] and does not operate in near real-time. Methods to improve computational efficiency is an area for future research. Nevertheless, to assess the feasibility of this optimal control approach for near-real-time implementation the design parameters in this evaluation intentionally reduce computational time even at a slight cost of calculating a less precise optimal control solution. For instance, the optimal control algorithm in this evaluation uses a global polynomial with 30 collocation nodes per 30-second time horizon. Based on the 1 Hz update rate and anticipated NAS intruder dynamics, this reduced number of collocation nodes appears appropriate for a receding horizon implementation especially since the nodes are tightly spaced at the beginning, which is the critical portion of the trajectory that the ownship plans to fly in the next time horizon. Further, when evaluating the conditional inequality constraint in equation (7.2) the optimal control algorithm uses a conservative exponential power of 24, which results in a maximum overestimation error of

72 feet in the horizontal and 24 feet in the vertical. These overestimation errors are slightly greater than typical sensor tolerances but very reasonable for a NAS collision avoidance application.

Based on the length of the time horizon and the algorithm update rate, a receding horizon implementation can present a challenge for an optimal control algorithm especially in a dynamic airborne collision avoidance environment where the subsequent time horizon could appear different than the proceeding time horizon due to the motion of the intruders. Although the sensors, radar and EO (in a noncooperative environment) and ADS-B and TCAS (in a cooperative environment), can sense beyond a 30-second time horizon, to be consistent with JOCA the optimization algorithm only optimizes against the next 30-second time horizon. For example, a scenario may have only one predicted intruder present in the first 30-second time horizon whereas subsequent 30-second time horizons may have additional intruders present. In this case, without a higher-level control algorithm guarding against suboptimal initial guesses, the predicted control sequence from the previous time horizon with only a single intruder may cause the optimization algorithm to converge to a local minimum. Implementing alternative control strategies to guard against non-desirable initial guesses is an area for future research. However, a recommendation which is evaluated to a limited-degree in the work herein is to extend the time horizon to include all potential intruders that could impact the optimal control solution when planning and executing an avoidance maneuver.

7.3 Simulation Methodology

For each simulation scenario, a high-fidelity optimal control solution is used to establish a baseline performance standard. To facilitate finding the global minimum, the optimal control method for these baseline evaluations consist of a single time horizon that extends to the time of closest point of approach (CPA) for the furthest intruder in the scenario. With this time horizon, unlike the fixed 30-second time horizon, the optimal control algorithm now has the complete time-history trajectory for all the intruders in the scenario, which increases the likelihood of finding the global optimal. Further, when calculating these

baseline solutions the optimal control algorithm uses a dense mesh of 200-collocation nodes distributed throughout the time horizon in 20-equally spaced intervals with 10-collocation nodes per interval. To further ensure a precise solution, when calculating the baseline optimal solution the optimal control algorithm evaluates the conditional inequality constraint in equation (7.2) using a power of 100 for the exponential (N) term, which results in a maximum overestimation error of 17.1 feet in the horizontal and 5.7 feet in the vertical. The results of these baseline evaluations then serve as a benchmark for comparing the collision avoidance trajectories produced by the JOCA algorithm and the optimal control algorithm developed in this research.

The evaluation criteria for these comparisons mirrors the cost functional shown in equation (7.1). For each scenario, the total cost for each algorithm is calculated by squaring the deviation distance in feet from the nominal trajectory in the horizontal (Δh_i) and vertical (Δv_i) directions at discrete one-second time increments starting at time zero (t_0) until the time to CPA (t_{CPA}) to the farthest intruder in the scenario. Like equation (7.1), these deviations are then normalized by the square of 3038 feet for horizontal deviations and 600 feet for vertical deviations. This cost formula appears as,

$$\text{Total Cost} = \sum_{i=t_0}^{t_{\text{CPA}}} \left[\left(\frac{\Delta h_i}{3038} \right)^2 + \left(\frac{\Delta v_i}{600} \right)^2 \right] \quad (7.9)$$

7.4 Scenario Description

This evaluation consists of four ownship-intruder scenarios. These scenarios are common collision avoidance test scenarios for the MIAA program and similar to the scenarios flown and evaluated during the airborne flight test program. At the start of each scenarios, the time to CPA to the closest intruder is 60 seconds. Likewise, the ownship's starting position is always at the origin and the intruder(s) start positions are based on a relative distance from the ownship. In Figures 7.1 - 7.4, adapted from the MIAA flight test program, the black dashed line represents the ownship's intended flightpath.

Scenario One is a single intruder scenario where the intruder is offset 0.2 NM north (y -axis) and 500 feet above (z -axis) the ownship. The intruder maintains a west heading of 180° and both the ownship and intruder maintain a constant speed of 300 ft/sec throughout the

scenario. The initial separation distance in the east direction (x -axis) between the ownship and intruder is 36,000 feet. Figure 7.1 graphically depicts the initial setup geometry for this scenario.

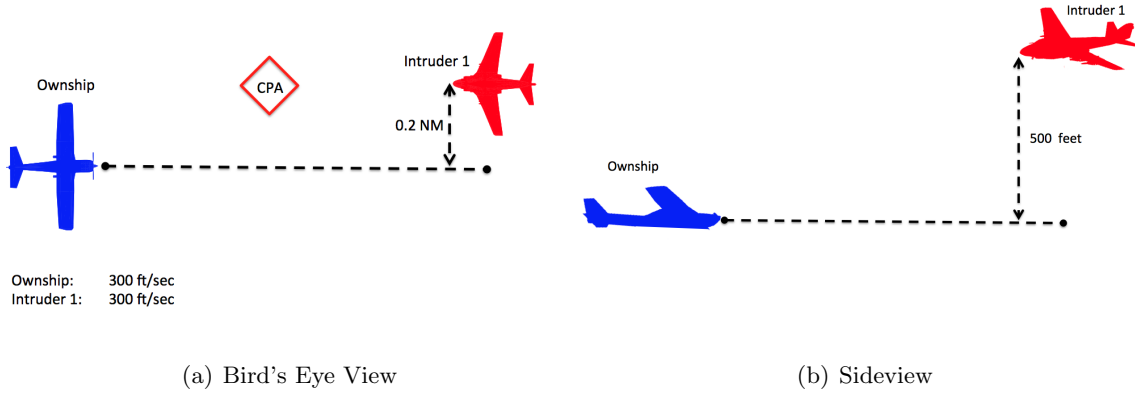


Figure 7.1: Scenario One

Scenario Two is identical to Scenario One but adds an additional intruder. The additional intruder in this scenario is offset 0.5 NM south and 300 feet above the ownship. Both intruders maintain a west heading of 180° and a constant speed of 300 ft/sec throughout the scenario. The second intruder in this scenario is 0.5 NM east of the first intruder. Based on a constant speed of 300 ft/sec for the intruders and ownship, the time to CPA to the second intruder is 70 seconds. Figure 7.2 graphically depicts the initial setup geometry for this scenario.

Scenario Three is a single intruder scenario where the intruder and ownship flightpaths cross perpendicular to one another. The intruder maintains a constant altitude and north heading of 90° . The ownship's intended flightpath is a constant 1250 feet/minute descent while maintaining an east heading. When the ownship crosses the CPA the intruder is 0.3 NM beyond the CPA and 300 feet below the CPA altitude. Both the ownship and intruder maintain a constant speed of 300 ft/sec throughout the scenario. Figure 7.3 graphically depicts the initial setup geometry for this scenario.

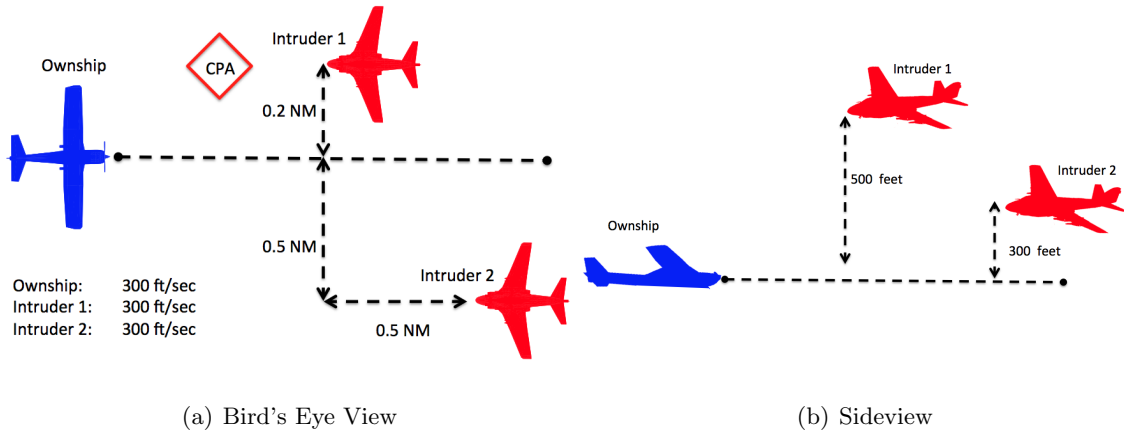


Figure 7.2: Scenario Two

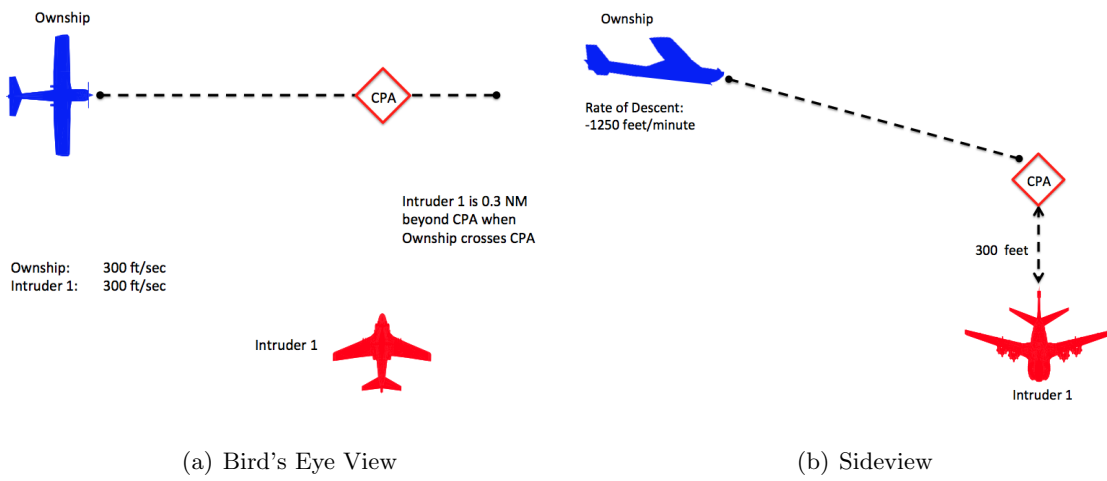


Figure 7.3: Scenario Three

Scenario Four is nearly identical to Scenario Three but adds an additional intruder. The additional intruder in this scenario is offset 0.5 NM east, 0.3 NM south, and 200 feet lower than Intruder 1. Both intruders maintain a north heading of 90° and a constant speed of 300 ft/sec throughout the scenario. Figure 7.4 graphically depicts the initial setup geometry for this scenario.

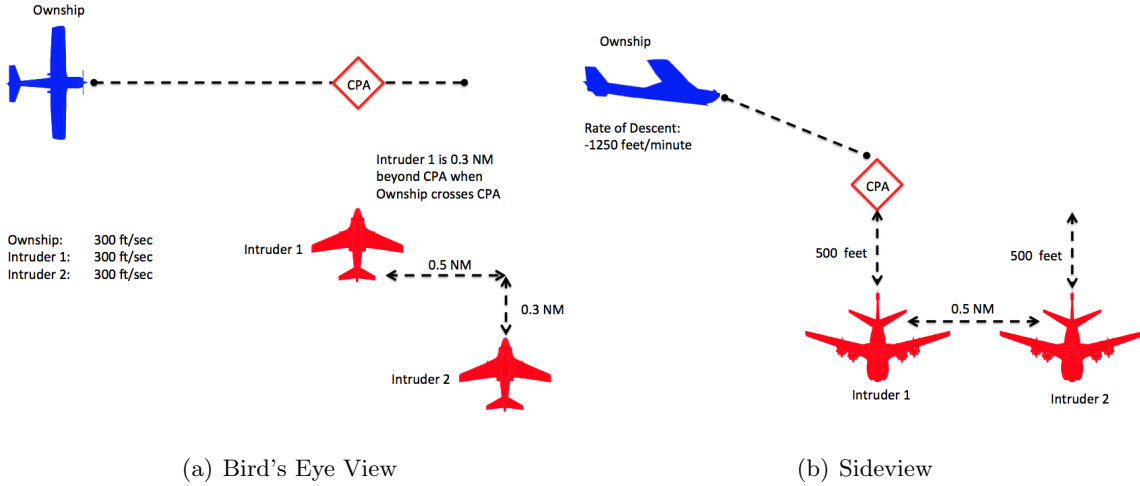


Figure 7.4: Scenario Four

7.5 Results

In the absence of a control penalty the optimal control solution naturally results in a bang-bang response between the upper and lower control limits to rapidly drive the cost functional to a minimum. However, in order to have a common cost to objectively compare performance results between JOCA and the optimal control approach, both the baseline and receding horizon optimal control solutions did not add a control weight penalty to the cost functional in equation (7.1). Most noticeably, without a control penalty the optimal control solution performs an S-turn in the horizontal-plane and a ‘porpoise’ in the vertical-plane about the nominal flightpath trajectory prior to commanding a maximum control maneuver away from the intruder(s). These slight S-turns or porpoise maneuvers enabled the optimal control solution to minimize the overall path deviation by remaining nearer to the nominal trajectory for a longer period of time. However, operationally these maneuvers can lead to excessive fuel consumption and degrade surveillance performance over a target area. Furthermore, these maneuvers are not what air traffic control or other pilots expect from an aircraft operating in the NAS. Therefore, for actual flight implementation a potential future research area is to appropriately scale and then quantify performance differences when applying a control penalty to minimize these oscillations about the nominal trajectory.

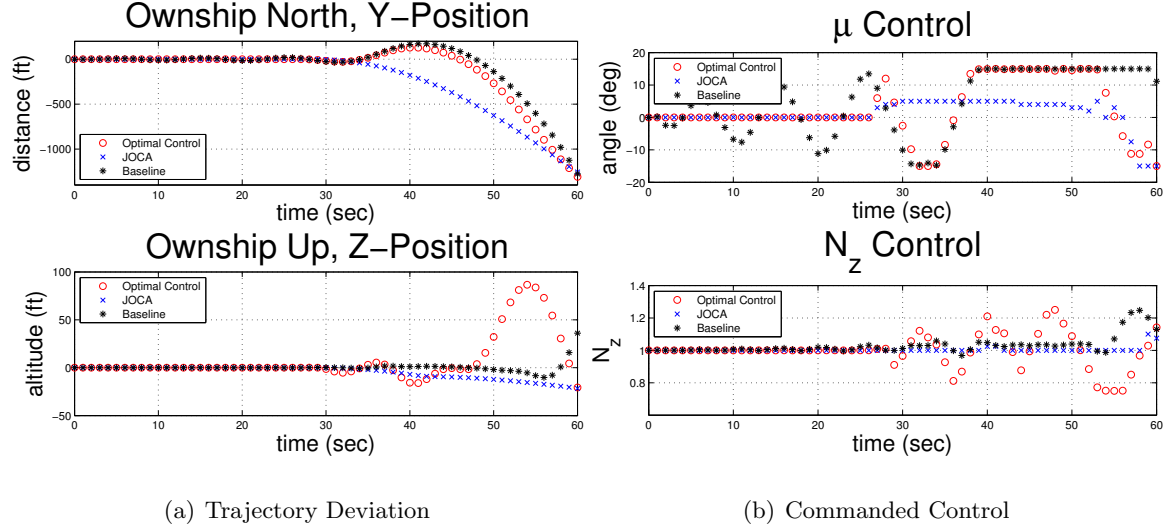


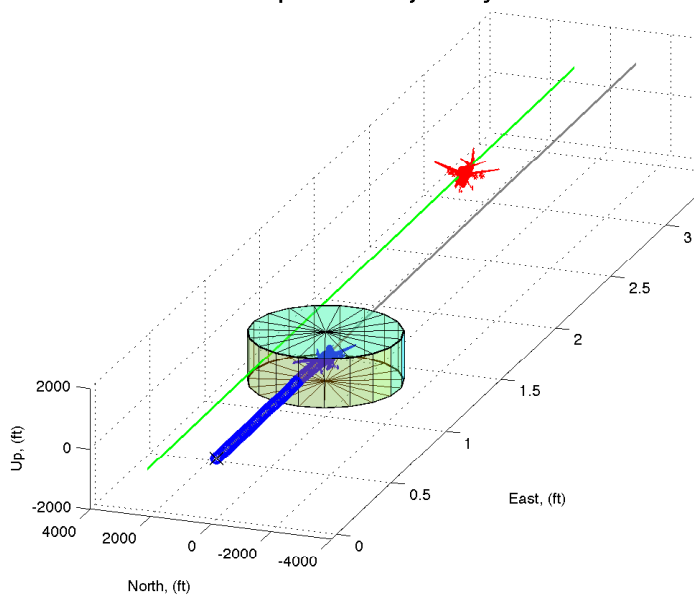
Figure 7.5: Results from Scenario One

7.5.1 Scenario One.

Figure 7.5 shows the horizontal and vertical trajectory deviations along with the control responses for the optimal control, JOCA, and baseline solutions for Scenario One. The optimal control solution is depicted with red circles, the JOCA solution with blue crosses, and the baseline solution with black asterisks. The lowest cost avoidance maneuver for this scenario was a level right turn away from the intruder. The basic shape for all three avoidance trajectories appeared similar. Based on equation (7.9), the total cost for the baseline solution was 2.97, the total cost for the optimal control solution was 3.60, and the total cost for the JOCA solution was 4.80. Thus, the optimal control solution was 21.2% greater than the baseline solution whereas the JOCA solution was 61.5% greater than the baseline solution. These differences are explained below.

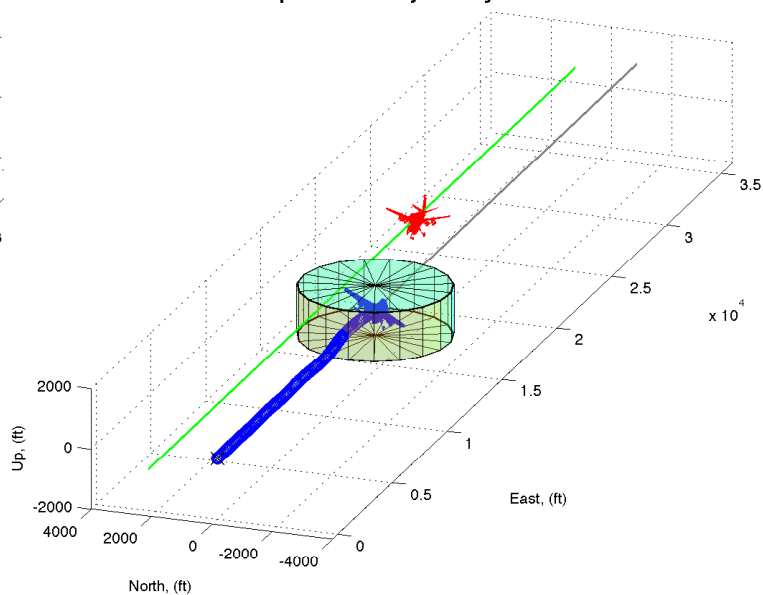
Figure 7.6 depicts the time-history trajectory for the baseline solution. In this figure the ownship is shown in blue and the intruder in red. The gray line represents the ownship's intended flightpath and the green line depicts the intruder's true flightpath. The cylinder around the ownship in this figure represents the conditional horizontal and vertical separation distances of 2460 feet in the horizontal and 820 feet in the vertical.

Baseline Optimal Trajectory 33.0 seconds



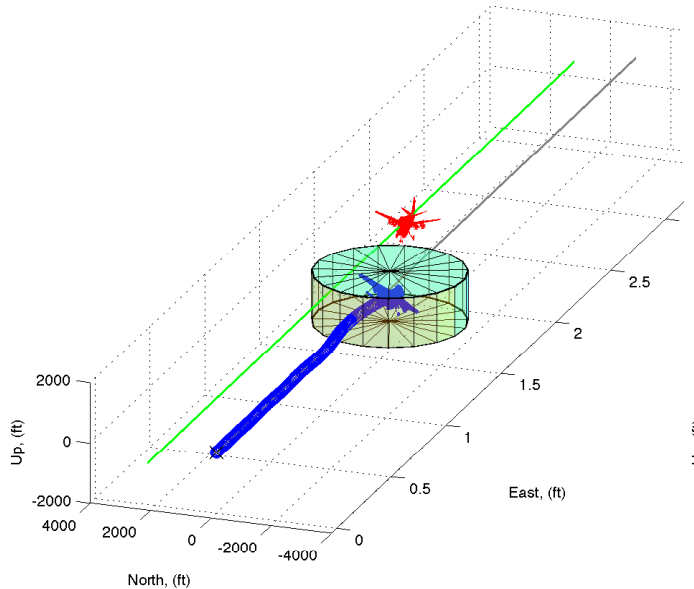
(a) Slight Turn into Intruder

Baseline Optimal Trajectory 48.0 seconds



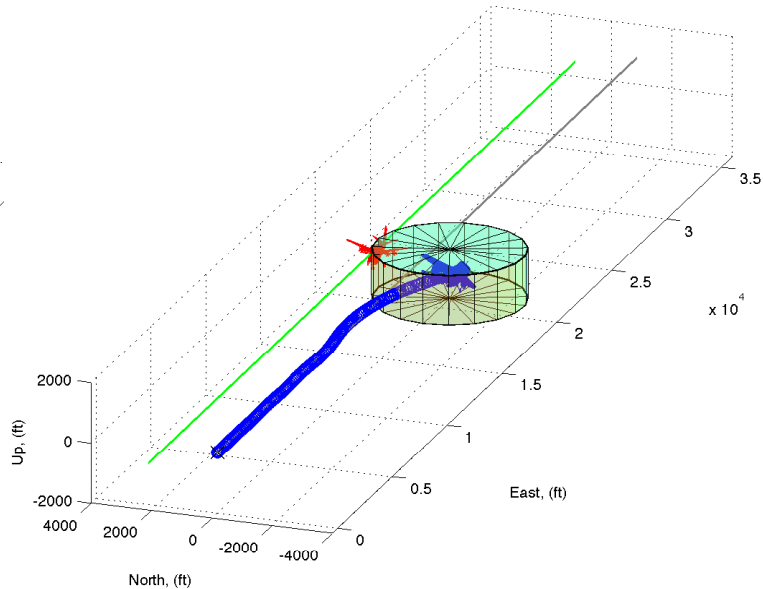
(b) Cross Nominal Trajectory in Turn Away from Intruder

Baseline Optimal Trajectory 51.0 seconds



(c) Continue Maximum Command Turn Away

Baseline Optimal Trajectory 60.0 seconds



(d) Satisfy Horizontal Separation Constraint at CPA

Figure 7.6: Scenario One - Time Series of Baseline Optimal Trajectory for Ownship (Blue) Avoiding Intruder (Red) While Minimizing Path Deviation.

In this scenario, the optimal control and baseline avoidance trajectories appeared very similar. At approximately 33 seconds the ownship began a slight S-turn into the intruder, effectively allowing the ownship to remain closer to the nominal flightpath longer prior to starting a maximum control bank away from the intruder at 40 seconds. At approximately 46 seconds for the optimal control solution and 48 seconds for the baseline solution the ownship departed the nominal trajectory in a maximum control right bank turn away from the intruder. However, a pronounced difference between the optimal control and baseline trajectories was the vertical profile. Due to the dynamics model in equation (7.5), a maximum control bank coupled with a maximum control normal acceleration would cause the ownship to climb. This fact coupled with an iterative plan-fly-plan receding horizon implementation of the optimal control problem amplified slight deviations from the nominal trajectory especially in the vertical direction since the initial control sequence guess for the next time horizon were the controls from the previous time horizon which had a slight climb at the start of the trajectory. Nonetheless, the resulting deviation in the vertical was only 86.6 feet which was relatively minor. Furthermore, this non-desirable deviation could be eliminated with an appropriate control penalty or feedback control algorithm.

The JOCA avoidance solution for this scenario, was a 5° right bank turn away from the intruder starting at 30 seconds. This shallower bank angle caused the JOCA solution to turn approximately 15 seconds earlier than the optimal control solution, which contributed to the larger total deviation cost. The JOCA avoidance solution also had a very minor descent which likely resulted from the slight bank without a corresponding increase in N_z .

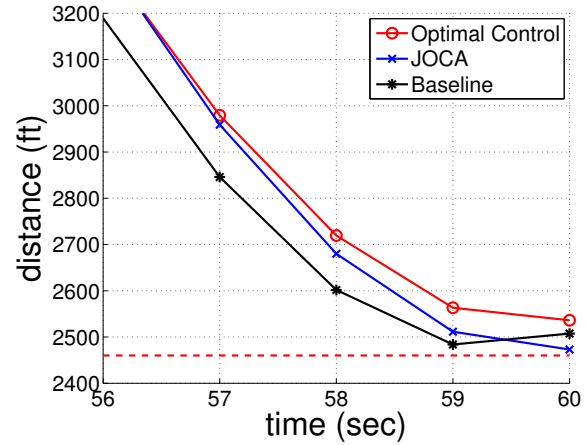


Figure 7.7: Distance from Intruder 1

Figure 7.7 shows the minimum horizontal separation distance for all three solutions. This figure only shows the last four seconds prior to CPA since this is the region where the

ownership was nearest to the intruder. The figure also does not include the vertical separation distances since all three solutions in this scenario remained within the conditional vertical separation constraint of 820 feet throughout the avoidance maneuver. In this scenario the JOCA solution achieved the closest minimum horizontal separation distance of 2473 feet from the intruder which occurred at 60 seconds. The minimum separation distance for the optimal control solution was 2536 feet which occurred at 60 seconds and 2484 feet for the baseline solution which occurred at 59 seconds.

7.5.2 Scenario Two.

Figure 7.8 shows the trajectory deviations and control responses for the optimal control, JOCA, and baseline solutions for Scenario Two. With only a single intruder, the previous

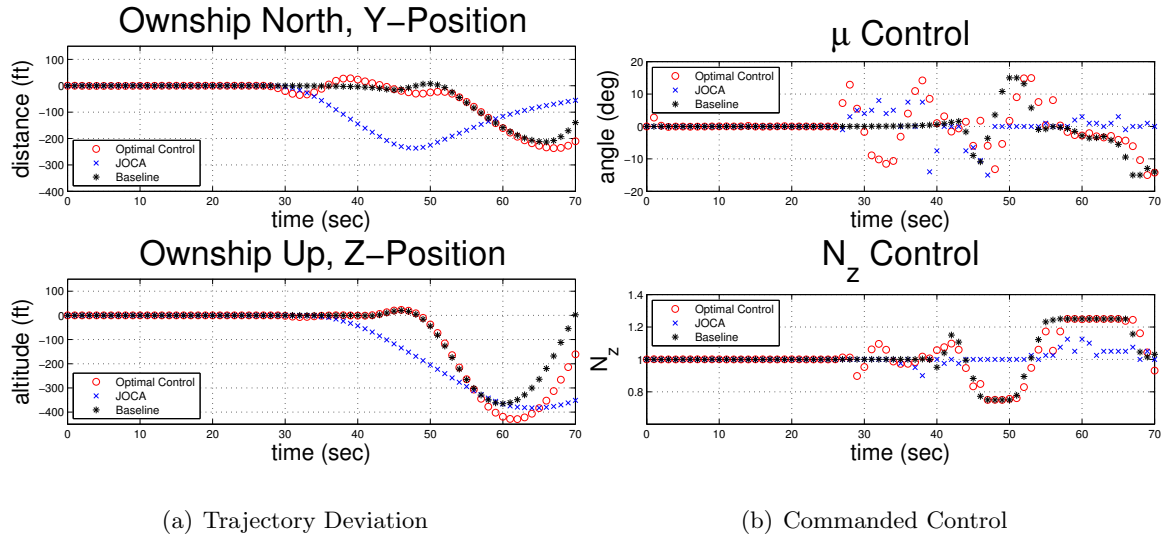


Figure 7.8: Results from Scenario Two

scenario demonstrated the least costly avoidance maneuver was a horizontal turn away. However, in the current scenario a horizontal avoidance maneuver away from Intruder 1 would no longer result in an overall optimal solution due to the presence of Intruder 2.

From equation (7.9), the total cost for the baseline solution for this scenario was 8.24, the total cost for the optimal control solution was 10.51, and the total cost for the JOCA solution was 13.84. The optimal control solution was 27.5% greater than the baseline

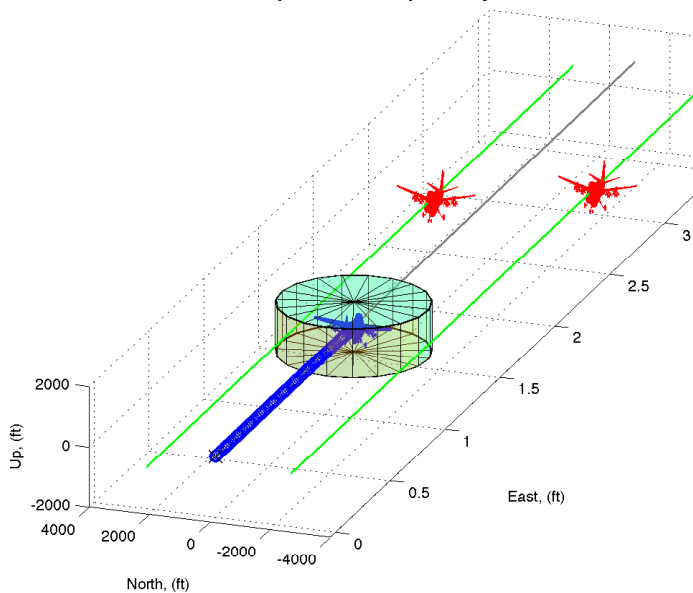
solution and the JOCA solution was 67.9% greater than the baseline solution. Like Scenario One, the lower cost for the optimal control solution resulted in part because the avoidance maneuver occurred later than the JOCA solution. Figure 7.9 on the next page shows the time-history trajectory for the baseline solution.

Based on the 30-second time horizon and the initial setup geometry, Intruder 1's predicted trajectory did not influence the avoidance solution until approximately 27 seconds into the scenario and Intruder 2's predicted trajectory did not enter the time horizon until approximately 5 seconds later. However, at 30 seconds into the scenario the JOCA algorithm began a horizontal turn away maneuver to avoid Intruder 1. At approximately 35 seconds the JOCA algorithm then began a descent to avoid Intruder 1 since a strictly turning avoidance maneuver would have violated the horizontal separation constraint for Intruder 2.

Like the JOCA algorithm, the optimal control solution did not have information about Intruder 2 at time 30 seconds. Therefore, similar to Scenario One, at approximately 30 seconds the optimal control solution began a slight S-turn about the nominal trajectory preparing to execute a maximum command turn away from Intruder 1. Yet, at approximately time 32 seconds Intruder 2 entered the time horizon and the optimal control algorithm could then calculate a new optimal control solution taking both intruders' predicted trajectories into account simultaneously. Based on this new concurrent optimization solution, the optimal control algorithm then transitioned from a strictly horizontal turn away to a combination of a vertical descent with a horizontal turn avoidance maneuver. The optimal solution started the descent at 48 seconds, which was 13 seconds after the JOCA algorithm started the descent. Likewise, the optimal control algorithm delayed the horizontal turn away maneuver until approximately 52 seconds, approximately 22 seconds after JOCA started the turn away.

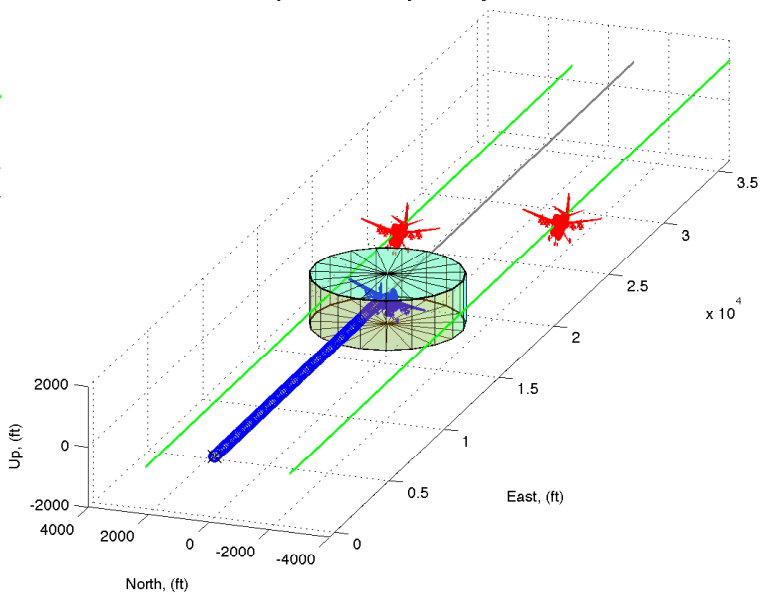
Figure 7.10 shows the minimum horizontal and vertical separation distances for all three solutions for both intruders. This figure shows the last 20 seconds prior to CPA for the second intruder. All three avoidance solutions appeared similar and caused the ownship to pass at least 820 feet underneath the first intruder just as the horizontal separation

Baseline Optimal Trajectory 42.0 seconds



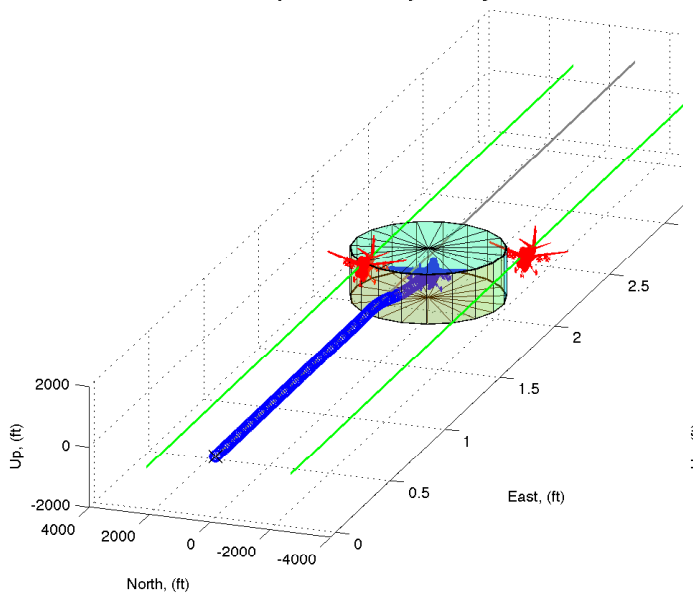
(a) No Avoidance Maneuver

Baseline Optimal Trajectory 52.5 seconds



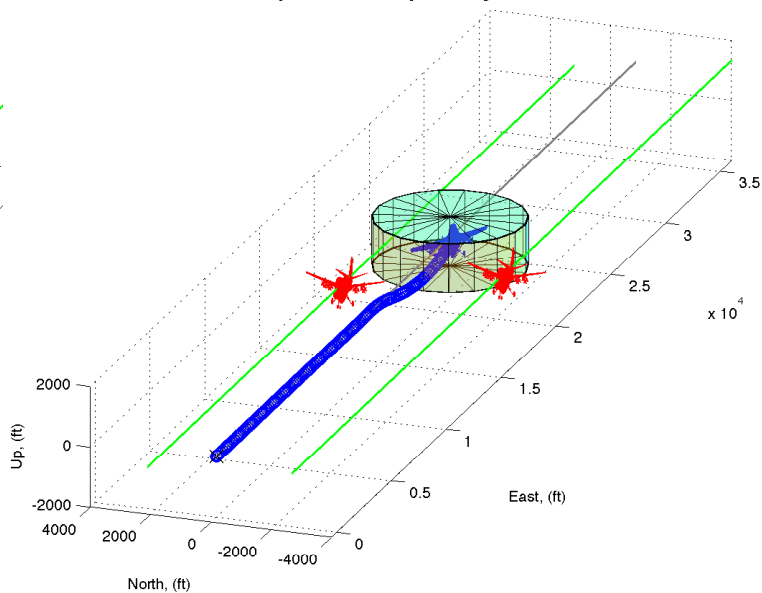
(b) Begin Vertical Descent and Slight Turn Away

Baseline Optimal Trajectory 63.0 seconds



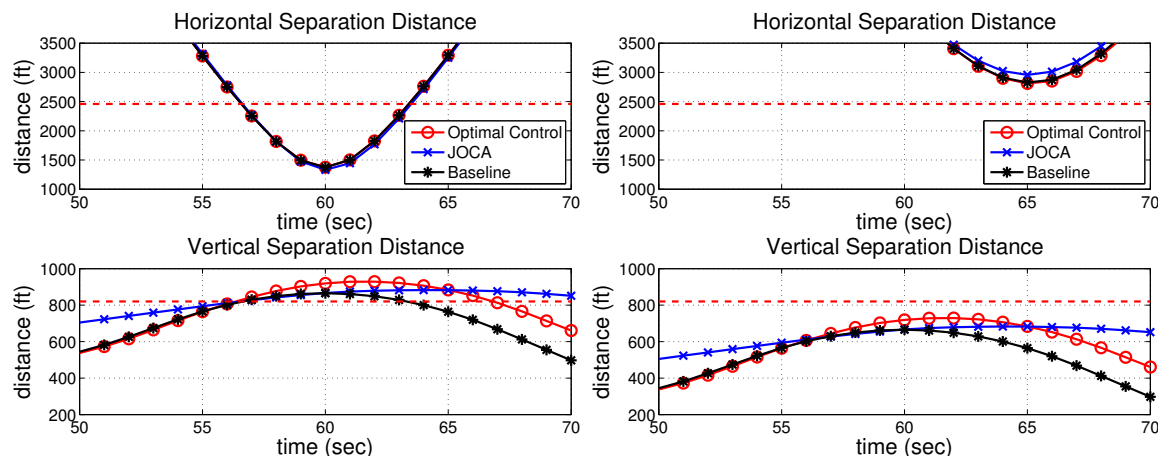
(c) Climb Back to Nominal

Baseline Optimal Trajectory 69.9 seconds



(d) Return to Course

Figure 7.9: Scenario Two - Time Series of Baseline Optimal Trajectory for Ownship (Blue) Avoiding Both Intruders (Red) While Minimizing Path Deviation.



(a) Intruder One

(b) Intruder Two

Figure 7.10: Separation Distance from Scenario Two

decreased below 2460 feet. As a limited investigation, this research evaluated Scenario Two for the optimal control algorithm using a time horizon that extended to 40 seconds. The additional 10 seconds allowed Intruder 2's predicted trajectory to be present in the time horizon throughout the avoidance maneuver. As a result, with this longer time horizon the optimal control solution approached closer to the baseline solution. The percentage difference from the baseline solution cost for this 40-second time horizon optimal control solution was 23.4% compared to a percentage difference of 27.5% with a 30-second time horizon. Although the savings were not significant in this scenario, a recommendation for further research is to conduct analysis to quantify the potential benefits of an adaptive time horizon which extends to include predicted trajectories for all intruders that could influence the avoidance solution.

7.5.3 Scenario Three.

Figure 7.11 on the following page shows the horizontal and vertical trajectory deviations along with the control responses for the optimal control, JOCA, and baseline solutions for Scenario Three. Figure 7.12 on page 144 shows the time-history trajectory for the baseline solution. Unlike the previous two scenarios, the JOCA trajectory differed significantly

from the baseline and optimal control avoidance trajectories. Although all three algorithms satisfied the conditional separation constraint, the JOCA solution continued the descent and performed a horizontal avoidance maneuver whereas the baseline and optimal control solutions arrested the descent and performed primarily a vertical avoidance maneuver.

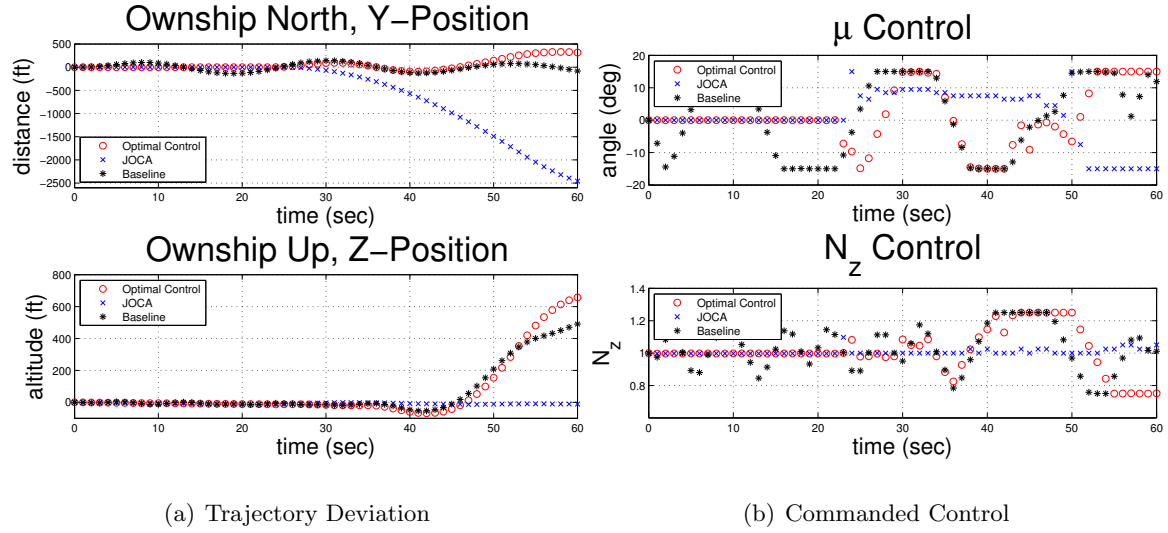


Figure 7.11: Results from Scenario Three

Figure 7.13 shows the minimum horizontal and vertical separation distance for all three solutions, including only from 50 seconds to the CPA since this was where the ownship was nearest to the intruder. From equation (7.9), the total cost for the baseline solution was 9.64, the total cost for the optimal control solution was 10.03, and the total cost for the JOCA solution was 11.44.

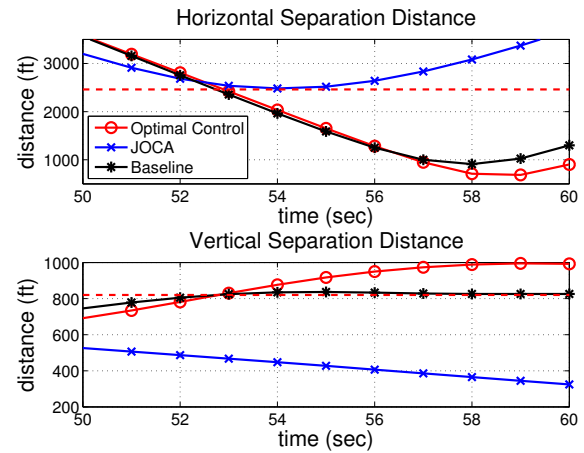
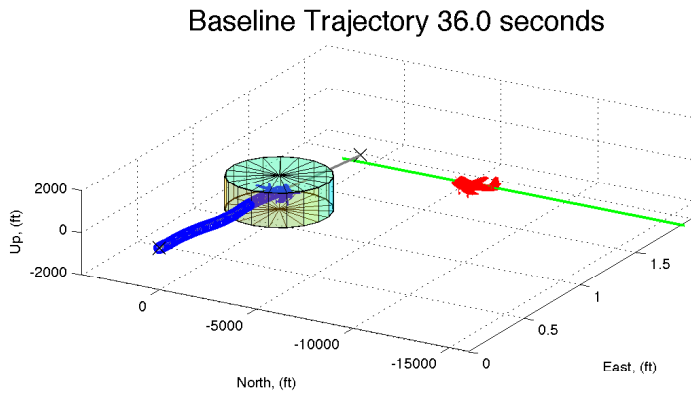
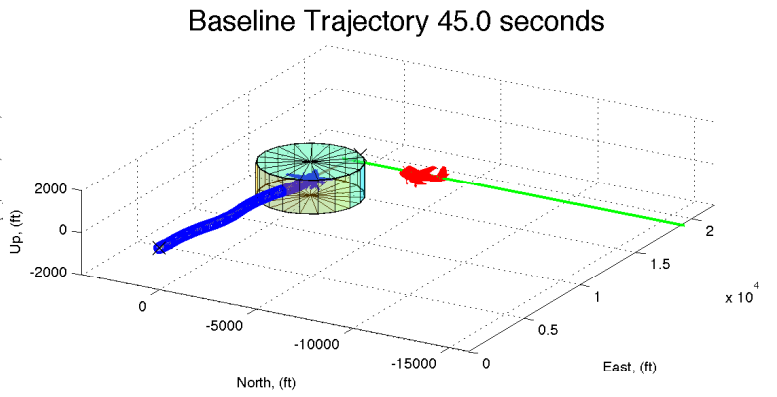


Figure 7.13: Distance from Intruder 1

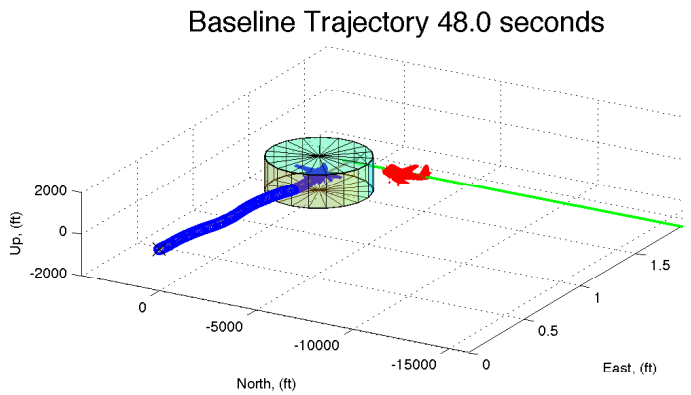
Thus, the optimal control solution was 3.95% greater than the baseline solution whereas the JOCA solution was 18.7% greater than the baseline solution. However, based on the results of the two previous scenarios where the canned or pre-set control levels commanded by the



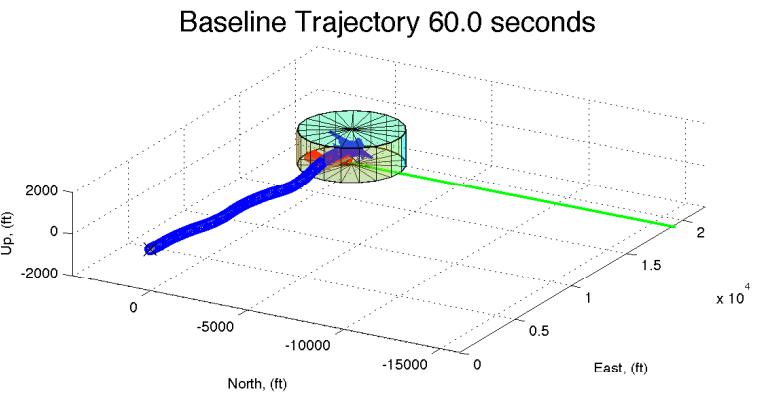
(a) In Descent with No Avoidance Maneuver



(b) Arrest Descent



(c) Climb with Slight Turn Towards Intruder



(d) Pass Above and Behind Intruder

Figure 7.12: Scenario Three - Time Series of Baseline Optimal Trajectory for Ownship (Blue) Avoiding Intruder (Red) While Minimizing Path Deviation.

JOCA algorithm caused the ownship to deviate sooner than necessary from the nominal trajectory, it seemed probable that the vertical avoidance maneuver was only less costly because the JOCA algorithm did not apply optimal controls in the horizontal avoidance maneuver. Therefore, to verify if the vertical solution was in fact less costly the baseline and optimal control solutions were artificially constrained so that they performed an avoidance maneuver which mirrored the JOCA solution; however, the total cost from a horizontal avoidance maneuver for both the baseline and optimal control solutions were slightly higher than their original vertical avoidance solutions. With a horizontal avoidance solution the total baseline cost was 10.24 and the total optimal control cost was 10.28. Thus, for this scenario the primarily vertical avoidance maneuver by the optimal control algorithm was less costly than the JOCA horizontal avoidance maneuver.

7.5.4 Scenario Four.

Figure 7.14 shows the trajectory deviations and control responses for the optimal control, JOCA, and baseline solutions for Scenario Four. Initially the same configuration was used as in the first three scenarios, but due to the limited number of collocation nodes and the steepness of the gradient, the exponential power of $N = 24$ for the conditional inequality path constraint in equation (7.2) for this scenario caused the optimizer to fail to converge. The optimizer successfully converged with an exponential power of $N = 16$ and these are the results shown in this section for the optimal control algorithm. Scenario Four was the most challenging of those tested due to the multi-intruders and ownship descent. This combination was more sensitive to the relative value of each of the gradients than in the previous scenarios with the same exponential power. Besides lowering the exponential power N for the conditional inequality path constraint in equation (7.2), another possible remedy for this situation would be to increase the number of collocation nodes.

Although the vertical trajectory deviations appeared similar for all three avoidance solutions, the JOCA solution arrested the descent and started the climb about 20 seconds earlier than the baseline and optimal control solutions. However, the baseline horizontal deviation trajectory differed markedly from the other two solutions. This difference was

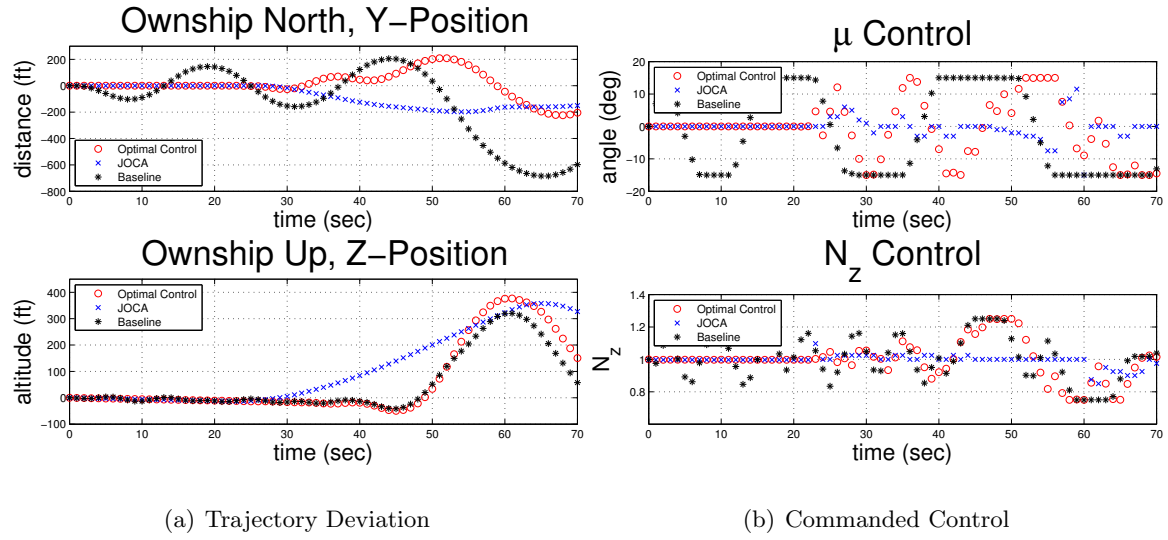


Figure 7.14: Results from Scenario Four

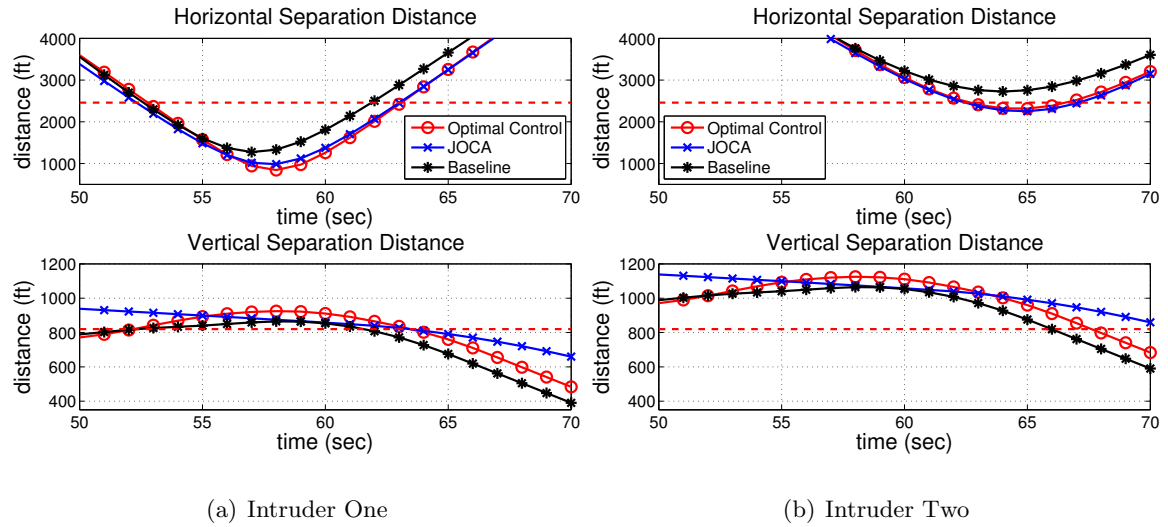
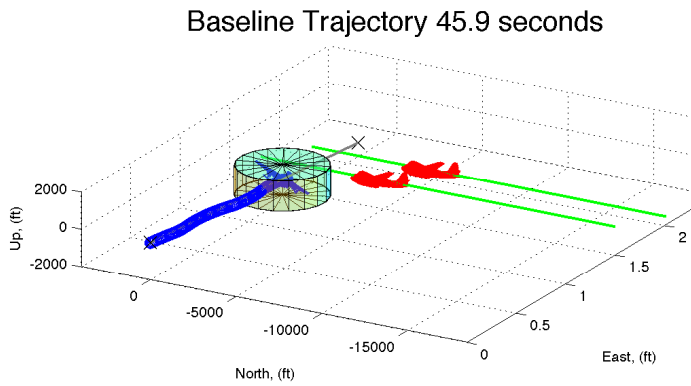
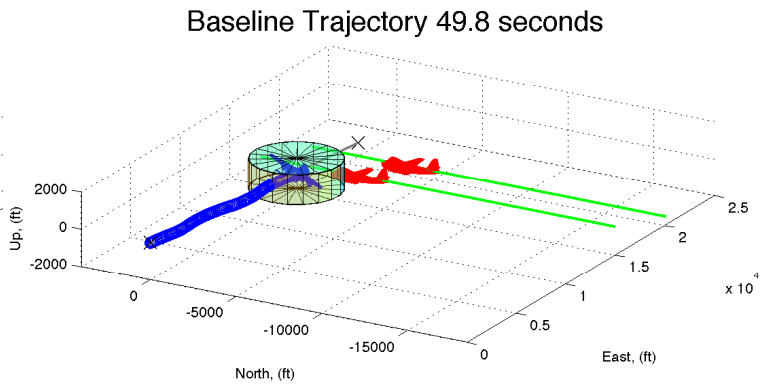


Figure 7.15: Separation Distance from Scenario Four

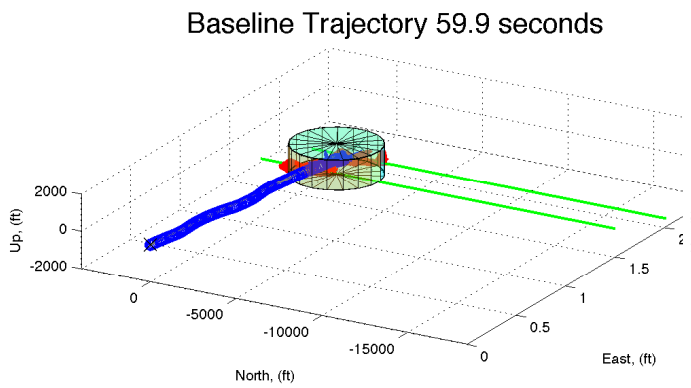
likely due to the longer planning horizon which allowed the baseline solution to account for both intruders throughout the entire avoidance maneuver. As a result, the baseline solution's larger horizontal deviation allowed the ownship to remain outside of the horizontal separation distance constraint for Intruder 2 and thereby continue on the descent profile



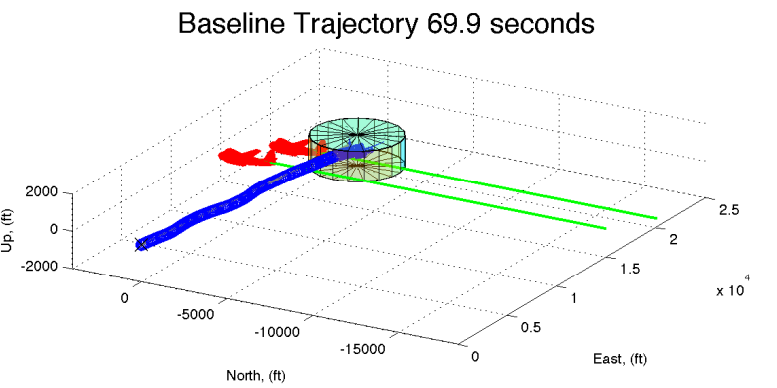
(a) Arrest Descent in Turn Towards Intruders



(b) Start Climb in Turn Toward Intruders



(c) Pass Above and Behind Intruder 1



(d) Return to Course

Figure 7.16: Scenario Four - Time Series of Baseline Optimal Trajectory for Ownship (Blue) Avoiding Intruders (Red) While Minimizing Path Deviation.

sooner as seen in Figure 7.15. From equation (7.9), the total cost for the baseline solution for this scenario was 10.93, the total cost for the optimal control solution was 11.04, and the total cost for the JOCA solution was 15.68. The optimal control solution was 1.0% greater than the baseline solution and the JOCA solution was 43.5% greater than the baseline solution. Figure 7.16 on the preceding page shows the time-history trajectory for the baseline solution.

7.6 Summary and Analysis of Results

Table 7.1 summarizes the results of the four avoidance scenarios. This table shows the percentage difference (increases) from the baseline solution for the optimal control and JOCA algorithms.

Table 7.1: Summary of Percentage Difference from Baseline Results

	Optimal Control Algorithm	JOCA Algorithm
Scenario One	21.2%	61.5%
Scenario Two	27.5%	67.9%
Scenario Three	3.95%	18.7%
Scenario Four	1.0%	43.5%

Using representative collision avoidance test scenarios for the MIAA program, this limited comparison showed that the optimal control algorithm produced a lower cost avoidance solution than the JOCA algorithm. Furthermore, in Scenario Three the optimal control solution identified a different avoidance trajectory that was more cost efficient than the JOCA solution. However, the optimal control problem offers no guarantee of finding the global minimum.

The JOCA algorithm on the other hand calculates 100 unique avoidance trajectories per 30-second time horizon. Because the algorithm applies pre-set control levels and not optimal controls, the JOCA avoidance trajectories are coarse solutions and not optimal with respect to the cost functional. Nonetheless, in general, because the algorithm searches 100 unique candidate trajectories the JOCA solution is more likely to identify the initial route of flight for the avoidance trajectory that is closer to the global minimum. Whereas comprehensive search algorithms such as genetic algorithms [102] and particle swarm optimization [103]

can increase the likelihood of finding the global minimum, in the context of this current research effort a possible means to achieve a “good” initial guess for the optimal control algorithm is to use the JOCA solution as an initial guess. This approach would utilize the strengths of both algorithms and likely result in a more cost efficient avoidance solution than either of these algorithms by themselves. This is a potential area for future research.

An even greater concern than guarding against a suboptimal initial guess is guarding against an infeasible initial guess, that is one that violates the inequality path constraints. Due to the measurement update rate and anticipated intruder dynamics in the NAS, the iterative nature of the receding horizon implementation inherently mitigates against an infeasible initial guess since the converged (feasible) solution from the previous time horizon is the initial guess for the current time horizon. However, circumstances such as a late sensor detection or an inopportune initial ownship-intruder starting geometry, can cause the initial straight-line guess to be infeasible. Using the JOCA solution as an initial guess is one method to prevent an infeasible initial guess. Another possible method to guard against an infeasible initial guess is addressed later in this document.

Another area which would benefit from additional research is to explore the differentiability (steepness) of the gradient for the conditional inequality path constraint in equation (7.2). This remains a particular concern especially when using a small number of fixed collocation nodes, which is likely required in order to satisfy a near real-time implementation. For example, in Scenario Four with only 30 fixed collocation nodes, an exponential power of $N = 24$ for the conditional inequality path constraint in equation (7.2) caused the optimizer to fail to converge; the optimizer successfully converged with an exponential power of $N = 16$. As stated in the earlier chapter on the development of conditional constraints for the work herein, an adaptive node placement strategy could alleviate this concern; however, an adaptive node placement algorithm is not conducive for real-time implementation since NLP convergence times for each time horizon is largely unknown. Thus, an analysis of efficient methods to mitigate against differentiability concerns remains an area for future research.

An additional area for future research is on methods to insure the differential constraints are satisfied when interpolating the state dynamics in an receding horizon implementation, which is necessary to establish the appropriate boundary conditions for the next time horizon. For instance, because the collocation nodes are not equally spaced the algorithm must interpolate the position (x , y , and z) and angle (γ and χ) states along with the states that model the actuator and system dynamics for the ownship's next one-second trajectory. Since the node spacing is dense at the beginning of the trajectory where the interpolation occurs, for low dynamic maneuvers this interpolation will likely not violate the differential constraints; however, in more dynamic cases the interpolated states could violate the differential constraints at the boundary condition because the optimizer only enforces equality at the collocation nodes. Enforcing boundary conditions could be done by solving the differential constraints analytically at the boundary conditions. Alternatively, loosening the bounds on the interpolated rate states would allow them to vary without violating the differential constraints. For this research this corresponded to opening the bounds on the highly dynamic control rate limit states (N_{z2} and μ_2) at the start of each time horizon.

As an example, in Scenario One near the CPA the optimizer results in a rapid change in the heading angle back to the nominal trajectory after the intruder passes. With the interpolated position states fixed, an upper and lower bound of ± 1 degree on the interpolated heading angle at this point in the trajectory allows the differential constraints at the beginning of the time horizon to be satisfied despite the rapidly changing and complex dynamics. Further, this upper and lower bound on the interpolated heading angle state does not violate any physical system constraints and the controls remain smooth and continuous throughout the trajectory. The results of this section show that even with rapidly changing and complex dynamics, loosening the bounds can prevent violating the differential constraints at the boundary conditions. Exploring methods and conditions to appropriately bound the interpolated states at the boundary conditions is a potential topic for future research.

Another potential topic for future research is the use of an adaptive time horizon which extends to include predicted trajectories for all intruders that could influence the avoidance solution. Although the limited evaluation in this section using an adaptive horizon did not show remarkable savings, there are potential collision avoidance scenarios where an adaptive time horizon could prove especially beneficial. One such example is where the initial avoidance maneuver is optimal for the present time horizon yet this same avoidance maneuver would expedite a potential collision with another intruder in a subsequent time horizon. The obvious downside to an adaptive horizon is the additional computational cost as well as the complexity of accounting for varying time horizons between measurement updates. Thus, a research effort to quantify the cost and benefits of an adaptive time horizon could prove beneficial.

Further, methods to improve NLP execution times are one of the principal considerations for enabling widespread use of an optimal control approach in real-time collision avoidance applications. Table 7.2 compares the the average, shortest, and longest NLP execution times for the receding horizon implementation in Scenario Two of this chapter. This table separates the time horizons into two categories: time horizons with intruders present and time horizons without intruders present.

Table 7.2: Summary of Optimal Control NLP Execution Times for Scenario Two

	With Intruders	Without Intruders
Average Time (seconds)	32.24	10.00
Shortest Time (seconds)	13.58	8.07
Longest Time (seconds)	51.64	12.08

The additional actuator and system dynamics states in equation (7.5) contributed to longer execution times as compared to previous evaluations. Further, as an additional reference, the NLP required 90.37 seconds to calculate the 200 collocation node baseline solution for Scenario One which had a 60-second time horizon and one intruder present. The NLP required 232.11 seconds for Scenario Two which had a 70-second time horizon and two intruders present. Methods to improve NLP execution times to facilitate near real-time implementation remains an area for future research. Likewise, in order to support

eventual flight implementation, a potential future research area is to quantify performance differences when applying a control penalty to minimize ownship oscillations about the nominal trajectory.

The comparison in this chapter between JOCA and the optimal control approach established the potential benefits of using an optimal control approach in an airborne sense and avoid application for the NAS with deterministic intruder models. Earlier chapters looked at the particle filter for estimating nonlinear intruder dynamics in a stochastic environment. The next chapter will analyze the performance of the particle filter against the more commonly used Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) techniques and compares their performance for use in an airborne sense and avoid application for the NAS.

VIII. Nonlinear Filter Development and Comparison

AS STATED previously, a fundamental necessity in solving the airborne collision avoidance problem is the need to estimate the current and future position of the intruder along with the need to accurately model how the probability regions associated with this position estimate change as a function of time. Previous work by the author [70] utilized a linear intruder model which did not take into account the natural correlation in x and y positions for a turning aircraft, and this model assumed a Gaussian distribution for the uncertainty probability regions associated with the predicted trajectory. An additional shortfall with this, and any linear model, is the inability to accurately estimate the future position of a turning aircraft. Therefore, this research utilizes a modified version of the coordinated turn model as presented in [83]. The coordinated turn model takes into account the natural correlation of x and y positions in the 2D horizontal plane for a turning aircraft and has the added benefit of estimating an intruder's turn-rate (ω), and thus, the model has the ability to estimate the future position of a turning aircraft. Clearly, for a predictive collision avoidance application the ability to estimate the future position for a turning aircraft is necessary. To this end, this chapter analyzes the simulation results of three different nonlinear filters and compares their performance for use in an airborne sense and avoid application within the National Airspace System (NAS). The motivation for this comparison is to address the question "How does the performance of these nonlinear filters compare when estimating and predicting an intruder's current and future trajectories for the optimal airborne collision avoidance problem?"

The ownship-intruder setup geometries for this evaluation are similar to the previous chapter where at the start of each 60-second scenario the two aircraft are either flying towards each other or flying at a 90° heading angle to one other. For each scenario the ownship and intruder maintains a constant speed of 300 ft/sec and the ownship flies a constant heading while the intruder performs either a standard-rate turn (3 deg/sec), a half-standard-rate turn (1.5 deg/sec), or no turn (linear propagation). In the first series of

scenarios the intruder remains co-altitude with the ownship. In the last series the intruder changes altitudes using a 1250 ft/min descent rate while still executing the different turn rates.

For this evaluation the sensor is a radar system located onboard the ownship aircraft and the sensor measurements are relative measurements between the ownship and the intruder that occur every second. Although an increased sampling rate would improve filter performance for all three filters, the 1 Hz update rate is used to help clearly distinguish their performance. The observation and dynamics models are described in the next section followed by the development of the nonlinear filters for this evaluation.

8.1 Observation and Dynamics Models for Filter Evaluation

The observation model is identical to the model described by equations (5.20) - (5.22); however, for convenience, this model is repeated here in equations (8.1) - (8.3). At each measurement time k , the algorithm simulates relative slant range (z_R), radial range rate (z_{vR}), azimuth (z_{az}), and elevation (z_{el}) measurements of the intruder with respect to the ownship. The measurement equations appear as [2]:

$$\mathbf{h}(\mathbf{x}(t)) = \begin{bmatrix} z_R \\ z_{vR} \\ z_{az} \\ z_{el} \end{bmatrix} = \begin{bmatrix} \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \\ \frac{\Delta x \Delta v_x + \Delta y \Delta v_y + \Delta z \Delta v_z}{\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}} \\ \tan^{-1} \frac{\Delta y}{\Delta x} \\ \sin^{-1} \frac{\Delta z}{\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}} \end{bmatrix} \quad (8.1)$$

where Δ represents the relative difference in position or velocity between the ownship and the intruder at time t . These measurements of the intruder are nonlinear functions, $\mathbf{h}(\mathbf{x}_k)$, of the intruder's state at time t_k and subject to noise, thus are modeled according to

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \quad (8.2)$$

where \mathbf{v}_k is zero-mean white Gaussian noise sequence modeled as,

$$\mathbf{E}[\mathbf{v}_j \mathbf{v}_k] = \mathbf{R}_m \delta_{jk} \quad (8.3)$$

where δ_{jk} is the Kronecker delta function. The noise autocorrelation, \mathbf{R}_m , represents the uncertainty associated with the sensor. The noise strength associated with each measurement appears along the diagonal elements of the matrix \mathbf{R}_m . For the work herein, the simulation uses the following noise autocorrelation matrix, which mirrors the values in the MIAA program [2]:

$$\mathbf{R}_m = \begin{bmatrix} (50 \text{ ft})^2 & 0 & 0 & 0 \\ 0 & (10 \text{ ft/sec})^2 & 0 & 0 \\ 0 & 0 & (1 \text{ deg})^2 & 0 \\ 0 & 0 & 0 & (0.8 \text{ deg})^2 \end{bmatrix} \quad (8.4)$$

As developed earlier, the dynamics model for this scenario consist of the 5-state coordinated turn model for the horizontal dimension (x, y) and the 3-state Singer acceleration model for the vertical dimension (z) . The states for this model are position (x, y, z) , velocity (v_x, v_y, v_z) , turn-rate (ω) , and vertical acceleration (a_z) . In continuous time, this model appears as:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{G}\mathbf{w}(t) = \begin{bmatrix} v_x(t) \\ v_y(t) \\ -\omega(t)v_y(t) \\ \omega(t)v_x(t) \\ -(1/\tau_\omega)\omega(t) \\ v_z(t) \\ a_z(t) \\ -(1/\tau_z)a_z(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{w}_\omega(t) \\ \mathbf{w}_a(t) \end{bmatrix} \quad (8.5)$$

where τ_ω is the maneuver time constant associated with the turn-rate (ω) state and τ_z is the maneuver time constant associated with the vertical acceleration (a_z) state. Further, $\mathbf{w}_\omega(t)$ and $\mathbf{w}_a(t)$ are zero mean white additive Gaussian noises with assumed

$$E[\mathbf{w}_\omega(t)\mathbf{w}_\omega^T(t+\tau)] = Q_\omega\delta(\tau) \quad \text{where} \quad Q_\omega = \frac{2\sigma_\omega^2}{\tau_\omega} \quad (8.6)$$

and,

$$E[\mathbf{w}_a(t)\mathbf{w}_a^T(t+\tau)] = Q_z\delta(\tau) \quad \text{where} \quad Q_z = \frac{2\sigma_z^2}{\tau_z} \quad (8.7)$$

In this evaluation, τ_ω was chosen as 30 seconds based on anticipated turn performance in the NAS and τ_z was chosen as 60 seconds based on [61]. The noise strength values are:

$$\sigma_\omega^2 = 1 \text{ deg}^2/\text{sec}^2 \quad (8.8)$$

$$\sigma_z^2 = 1 \text{ ft}^2/\text{sec}^4 \quad (8.9)$$

The value in equation (8.8) is based on work by [84]. The value in equation (8.9) is from experimentation since a historical database like the aircraft turn-rate histograms in [84] existed only for overall aircraft acceleration and not just for vertical acceleration. Thus, the initial mean (\mathbf{x}_0) appears as,

$$\mathbf{x}_0 = [x_{0\text{true}} \ y_{0\text{true}} \ v_{x0\text{true}} \ v_{y0\text{true}} \ 0 \ z_{0\text{true}} \ v_{z0\text{true}} \ 0]^T + \mathbf{\Gamma}^T \mathbf{w} \quad (8.10)$$

and the initial covariance (\mathbf{P}_0) appears as,

$$\mathbf{P}_0 = \begin{bmatrix} (25 \text{ ft})^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & (25 \text{ ft})^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & (2 \text{ ft/sec})^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & (2 \text{ ft/sec})^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & (1 \text{ deg/sec})^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & (50 \text{ ft})^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & (2 \text{ ft/sec})^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & (1 \text{ ft/sec}^2)^2 \end{bmatrix}$$

where \mathbf{w} is normally distributed, $\sim \mathcal{N}(0, 1)$, and $\mathbf{\Gamma}$ represents the Cholesky decomposition of the initial covariance matrix \mathbf{P}_0 such that,

$$\mathbf{P}_0 = \mathbf{\Gamma}^T \mathbf{\Gamma} \quad (8.11)$$

As in [104], the noise corrupted observations are generated from the truth data by adding random noise from a zero-mean Gaussian distribution. In order to standardize the evaluation, for each scenario the identical sequence of noise corrupted observations are used for each filter. Likewise, at the start of each scenario the same set of random numbers for all three filters is used to generate the initial mean value, \mathbf{x}_0 , in equation (8.10). The next section describes the filters used in this evaluation.

8.2 Filter Description

The Kalman filter [105] is an optimal estimator when using purely linear dynamics models with additive white Gaussian noise. At each time step, the recursive “Kalman filter assumes the posterior density is Gaussian and hence exactly and completely characterized by two parameters, its means and covariance” [6]. The recursive equations for the Kalman filter are important to establish the derivation of the nonlinear estimation filters used in this research. Therefore, these equations appear as follows:

Given the linear dynamics models:

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) + \mathbf{G}(t)\mathbf{w}(t) \quad (8.12)$$

the Kalman filter assumes a linear discrete-time observation model that appears as:

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \quad (8.13)$$

where the discrete-time state transition matrix appears as:

$$\Phi(t_{k-1}, \tau) = e^{\mathbf{F}\tau} \quad (8.14)$$

$$= \Phi_{k-1}(\Delta t) \quad (8.15)$$

where τ and Δt are equivalently the time difference between t_{k-1} and t_k . The *a priori* state estimate and covariance appear as:

$$\hat{\mathbf{x}}_k^- = \Phi(t_{k-1}, \tau) \hat{\mathbf{x}}_{k-1}^+ + \mathbf{B}_{k-1} \mathbf{u}_{k-1} \quad (8.16)$$

$$\hat{\mathbf{P}}_k^- = \Phi(t_{k-1}, \tau) \hat{\mathbf{P}}_{k-1}^+ \Phi(t_{k-1}, \tau)^T + \mathbf{Q}_{k-1} \quad (8.17)$$

and the Kalman filter recursive equations appear as:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T [\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k]^{-1} \quad (8.18)$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k [\mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_k^-] \quad (8.19)$$

$$\mathbf{P}_k^+ = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_k^- \quad (8.20)$$

where \mathbf{Q}_{k-1} in equation (8.17) is the discrete-time noise strength which appears as:

$$\mathbf{Q}_{k-1} = \int_{t_{k-1}}^{t_k} \boldsymbol{\Phi}(t_{k-1}, \tau) \mathbf{G} \mathbf{Q}_c \mathbf{G}^T \boldsymbol{\Phi}(t_{k-1}, \tau)^T d\tau \quad (8.21)$$

and \mathbf{Q}_c is the power spectral density of the zero mean white additive Gaussian noise such that:

$$E[\mathbf{w}(t) \mathbf{w}^T(t + \tau)] = \mathbf{Q}_c \delta(\tau) \quad (8.22)$$

Although equation (8.21) is often difficult to solve analytically, one efficient method to solve this equation is using the Van Loan method [106, 107]. Since the dynamics model in this research are nonlinear, the derivation of the Van Loan method is shown next for a nonlinear system.

8.2.1 Van Loan Method.

Given a nonlinear continuous time system of the form

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{f}(\mathbf{x}(t)) + \mathbf{G} \mathbf{w}(t) \quad (8.23)$$

where \mathbf{G} is a time-invariant matrix and $\mathbf{w}(t)$ is zero mean white additive Gaussian noise with power spectral density, \mathbf{Q}_c , as defined in equation (8.22). In certain cases, such as in equations (5.18) and (5.19) based on the Singer acceleration model, the discrete-time noise model, \mathbf{Q}_k , can be determined analytically; however, in cases where this is not possible \mathbf{Q}_k can still be calculated efficiently using the following Van Loan matrix fraction decomposition which appears as [106, 107]:

$$\exp \left\{ \begin{bmatrix} -\mathbf{F} & \mathbf{G} \mathbf{Q}_c \mathbf{G}^T \\ \mathbf{0} & \mathbf{F}^T \end{bmatrix} t \right\} = \begin{bmatrix} \mathbf{D}_{11}(t) & \mathbf{D}_{12}(t) \\ \mathbf{0} & \mathbf{D}_{22}(t) \end{bmatrix} \quad (8.24)$$

where \mathbf{F} is the Jacobian matrix of $\mathbf{f}(\mathbf{x}(t))$,

$$\mathbf{F} \triangleq \left. \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\mathbf{x}}} \quad (8.25)$$

thus, from equation (8.24) the discrete-time noise strength \mathbf{Q}_k equals:

$$\mathbf{Q}_k = \mathbf{D}_{22}^T \mathbf{D}_{12} \quad (8.26)$$

Proof: The full proof of the Van Loan method appears in [106]. Based on [106], an abbreviated proof specific to this application is shown below for completeness:

Given a $n \times n$ block triangular matrix \mathbf{B} defined as:

$$\mathbf{B} = \begin{bmatrix} -\mathbf{A} & \mathbf{P} \\ \mathbf{0} & \mathbf{A}^T \end{bmatrix} \text{ then for } t \geq 0 \quad (8.27)$$

$$e^{\mathbf{B}t} = \begin{bmatrix} \mathbf{D}_{11}(t) & \mathbf{D}_{12}(t) \\ \mathbf{0} & \mathbf{D}_{22}(t) \end{bmatrix} \quad (8.28)$$

where

$$e^{\mathbf{B}t} \Big|_{t=0} = \mathbf{I} = \begin{bmatrix} \mathbf{D}_{11}(0) & \mathbf{D}_{12}(0) \\ \mathbf{0} & \mathbf{D}_{22}(0) \end{bmatrix} \quad (8.29)$$

Therefore,

$$\begin{aligned} \mathbf{D}_{11}(0) &= \mathbf{I} \\ \mathbf{D}_{12}(0) &= \mathbf{0} \\ \mathbf{D}_{22}(0) &= \mathbf{I} \end{aligned} \quad (8.30)$$

Taking the derivative of equation (8.28) results in:

$$\begin{aligned} \frac{d}{dt} e^{\mathbf{B}t} &= \mathbf{B} e^{\mathbf{B}t} = \begin{bmatrix} \dot{\mathbf{D}}_{11}(t) & \dot{\mathbf{D}}_{12}(t) \\ \mathbf{0} & \dot{\mathbf{D}}_{22}(t) \end{bmatrix} \\ &= \begin{bmatrix} -\mathbf{A} & \mathbf{P} \\ \mathbf{0} & \mathbf{A}^T \end{bmatrix} \begin{bmatrix} \mathbf{D}_{11}(t) & \mathbf{D}_{12}(t) \\ \mathbf{0} & \mathbf{D}_{22}(t) \end{bmatrix} \\ &= \begin{bmatrix} -\mathbf{A}\mathbf{D}_{11} & -\mathbf{A}\mathbf{D}_{12} + \mathbf{P}\mathbf{D}_{22} \\ \mathbf{0} & \mathbf{A}^T \mathbf{D}_{22} \end{bmatrix} \end{aligned} \quad (8.31)$$

where

$$\dot{\mathbf{D}}_{11}(t) = -\mathbf{A}\mathbf{D}_{11} \quad \mathbf{D}_{11}(0) = \mathbf{I} \quad (8.32)$$

$$\dot{\mathbf{D}}_{22}(t) = \mathbf{A}^T \mathbf{D}_{22} \quad \mathbf{D}_{22}(0) = \mathbf{I} \quad (8.33)$$

$$\dot{\mathbf{D}}_{12}(t) = -\mathbf{A}\mathbf{D}_{12} + \mathbf{P}\mathbf{D}_{22} \quad \mathbf{D}_{12}(0) = \mathbf{0} \quad (8.34)$$

Therefore,

$$\mathbf{D}_{11} = e^{-\mathbf{A}t} \quad (8.35)$$

$$\mathbf{D}_{22} = e^{\mathbf{A}^T t} \quad (8.36)$$

Substituting equation (8.36) into equation (8.34) results in,

$$\dot{\mathbf{D}}_{12}(t) = -\mathbf{A}\mathbf{D}_{12} + \mathbf{P}e^{\mathbf{A}^T t} \quad \mathbf{D}_{12}(0) = \mathbf{0} \quad (8.37)$$

Next, assuming a form of the solution for \mathbf{D}_{12} where

$$\mathbf{D}_{12} = e^{-\mathbf{A}t} \mathbf{U}(t) \quad \mathbf{D}_{12}(0) = \mathbf{I}\mathbf{U}(0) = \mathbf{0} \quad (8.38)$$

Thus,

$$\dot{\mathbf{D}}_{12} = -\mathbf{A}e^{-\mathbf{A}t} \mathbf{U}(t) + e^{-\mathbf{A}t} \dot{\mathbf{U}}(t) \quad (8.39)$$

Combining equations (8.37) and (8.38) gives

$$\dot{\mathbf{D}}_{12} = -\mathbf{A}e^{-\mathbf{A}t} \mathbf{U}(t) + \mathbf{P}e^{\mathbf{A}^T t} \quad (8.40)$$

where the equality in equations (8.39) and (8.40) results in

$$e^{-\mathbf{A}t} \dot{\mathbf{U}}(t) = \mathbf{P}e^{\mathbf{A}^T t} \quad (8.41)$$

$$\dot{\mathbf{U}}(t) = e^{\mathbf{A}t} \mathbf{P}e^{\mathbf{A}^T t} \quad (8.42)$$

Therefore,

$$\mathbf{U}(t) = \int_0^t e^{\mathbf{A}\tau} \mathbf{P}e^{\mathbf{A}^T \tau} d\tau \quad (8.43)$$

where,

$$\mathbf{D}_{12}(t) = e^{-\mathbf{A}t} \mathbf{U}(t) \quad (8.44)$$

$$= e^{-\mathbf{A}t} \int_0^t e^{\mathbf{A}\tau} \mathbf{P} e^{\mathbf{A}^T \tau} d\tau \quad (8.45)$$

and since,

$$\mathbf{D}_{22}^T(t) = e^{\mathbf{A}t} \quad (8.46)$$

Therefore,

$$\mathbf{D}_{22}^T \mathbf{D}_{12} = \int_0^t e^{\mathbf{A}\tau} \mathbf{P} e^{\mathbf{A}^T \tau} d\tau \quad \square \quad (8.47)$$

8.2.2 EKF Development.

For the nonlinear dynamics described by equation (8.23), the EKF linearizes about the nominal estimate $\hat{\mathbf{x}}$ by taking the Jacobian as defined in equation (8.25) of the dynamics function $\mathbf{f}(\mathbf{x}(t))$. Based on the dynamics model in equation (8.5), the Jacobian matrix for the dynamics function $\mathbf{f}(\mathbf{x}(t))$ in this evaluation appears as,

$$\mathbf{F} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\omega & -v_y & 0 & 0 & 0 \\ 0 & 0 & \omega & 0 & v_x & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -(1/\tau_\omega) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -(1/\tau_z) \end{bmatrix} \quad (8.48)$$

where the discrete-time state transition matrix is calculated using equation (8.14). Likewise, for the nonlinear observation model described by equation (8.2), the EKF linearizes about the nominal estimate $\hat{\mathbf{x}}$ by taking the Jacobian of the measurement function $\mathbf{h}(\mathbf{x}_k)$ where \mathbf{H} is defined as,

$$\mathbf{H} \triangleq \left. \frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\mathbf{x}}} \quad (8.49)$$

Based on the nonlinear observation model in equation (8.1), the Jacobian matrix for the measurement function $\mathbf{h}(\mathbf{x}_k)$ in this evaluation appears as,

$$\mathbf{H} = \begin{bmatrix} \frac{\Delta x}{\rho} & \frac{\Delta y}{\rho} & 0 & 0 & 0 & \frac{\Delta z}{\rho} & 0 & 0 \\ \frac{\Delta v_x \rho_{yz}^2 - \Delta x(\varrho_{yz})}{\rho^3} & \frac{\Delta v_y \rho_{xz}^2 - \Delta y(\varrho_{xz})}{\rho^3} & \frac{\Delta x}{\rho} & \frac{\Delta y}{\rho} & 0 & \frac{\Delta v_z \rho_{xy}^2 - \Delta z(\varrho_{xy})}{\rho^3} & \frac{\Delta z}{\rho} & 0 \\ -\frac{\Delta y}{\rho_{xy}^2} & \frac{\Delta x}{\rho_{xy}^2} & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{\Delta x \Delta z}{\rho_{xy} \rho^2} & -\frac{\Delta y \Delta z}{\rho_{xy} \rho^2} & 0 & 0 & 0 & \frac{\rho_{xy}}{\rho^2} & 0 & 0 \end{bmatrix}$$

where

$$\begin{aligned} \rho &= \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \\ \rho_{xy} &= \sqrt{\Delta x^2 + \Delta y^2} \\ \rho_{yz} &= \sqrt{\Delta y^2 + \Delta z^2} \\ \rho_{xz} &= \sqrt{\Delta x^2 + \Delta z^2} \end{aligned} \tag{8.50}$$

and

$$\begin{aligned} \varrho_{xy} &= \Delta x \Delta v_x + \Delta y \Delta v_y \\ \varrho_{yz} &= \Delta y \Delta v_y + \Delta z \Delta v_z \\ \varrho_{xz} &= \Delta x \Delta v_x + \Delta z \Delta v_z \end{aligned} \tag{8.51}$$

8.2.3 UKF Development.

In the UKF the mean $\bar{\mathbf{x}}$ and covariance \mathbf{P}_x are approximated by $2L+1$ weighted points where L is the number of states [108, 109]. The UKF equations from [108, 109] are repeated here for convenience and appear as,

$$\begin{aligned}
\chi_0 &= \bar{\mathbf{x}} \\
\chi_i &= \bar{\mathbf{x}} + \left(\sqrt{(L + \lambda) \mathbf{P}_{\mathbf{x}}} \right)_i \quad i = 1, \dots, L \\
\chi_i &= \bar{\mathbf{x}} - \left(\sqrt{(L + \lambda) \mathbf{P}_{\mathbf{x}}} \right)_{i-L} \quad i = L + 1, \dots, 2L \\
\mathbf{W}_0^m &= \frac{\lambda}{(L + \lambda)} \\
\mathbf{W}_0^c &= \frac{\lambda}{(L + \lambda) + (1 - \alpha^2 + \beta)} \\
\mathbf{W}_i^m = \mathbf{W}_i^c &= \frac{1}{\{2(L + \lambda)\}} \quad i = 1, \dots, 2L
\end{aligned} \tag{8.52}$$

where $\lambda = \alpha^2(L + \kappa) - L$ is used as a scaling parameter for the transformation [109]. The parameter α determines the spread of the sigma points about the mean, and in practice this value is often set to a small positive number [109]. For this evaluation, $\alpha = 0.001$. Likewise, in accordance with the usual practice [109], the secondary scaling parameter κ is set to zero for this evaluation. The parameter β is used to account for prior knowledge of the distribution for the random variable \mathbf{x} . If the distribution is Gaussian, then $\beta = 2$ is the optimal value [109]. In the absence of prior knowledge about the distribution, β is often set to a value of two, which is the case for this evaluation. Finally, the discrete-time noise strength \mathbf{Q}_k is calculated using the Van Loan method described by equations (8.24) and (8.26) where the \mathbf{F} matrix is calculated using equation (8.48) with the linearization about $\bar{\mathbf{x}}$.

8.2.4 PF Development.

Since the PF has already been implemented earlier in this document, this section highlights differences in the implementation described in this chapter from the implementation described in previous chapters. These differences include: threshold resampling to improve sample diversity, use of a more efficient resampling algorithm, updates to the discrete-time noise calculation, and the addition of numerical differentiation. A brief description follows of these differences starting with threshold resampling.

Resampling at every measurement update can cause a loss of particle diversity [6] and lead to collapse of the filter especially when using a low process noise input. One method

to preserve diversity and prevent filter collapse is to only resample when the number of effective particles (N_{eff}) is less than a user-defined threshold value, (N_{thr}) where,

$$N_{\text{eff}} = \frac{1}{\sum_{i=1}^N (w_k^i)^2} < N_{\text{thr}}. \quad (8.53)$$

where N is the number of particles and w_k^i is the particle weight at time k . The values for N_{thr} typically range between $\frac{N}{2} < N_{\text{thr}} < \frac{2N}{3}$. In this evaluation the threshold is chosen as $N_{\text{thr}} = \frac{2N}{3}$ based on [44]. In addition to preserving particle diversity, another benefit of not resampling after every measurement update is the filter's computational efficiency improves, which is advantageous for any PF application with a goal of near real-time implementation. Implementing an efficient resampling algorithm is key to improving computational efficiency.

Figure 8.1 from [6, 7] shows a single cycle of the particle filter for a SIR implementation where the resampling step produces uniformly weighted estimates using an approximation of the posterior distribution. Efficient resampling algorithms exist in the literature such as Ripley's resampling algorithm as shown in [110] or others in [110] that perform this resampling step without the use of computationally costly '*for*' loops as used earlier in this research. The CPU efficiency metric for comparing the three different nonlinear filters in this section highlights the performance gains in the PF using a more efficient resampling algorithm as compared to the baseline resampling algorithm implemented in the earlier chapters. In addition to efficiency, the ability to accurately model system dynamics is important for filter implementation.

In the earlier PF implementation, the calculation for the discrete-time noise strength \mathbf{Q}_k mirrored the formulation in the MIAA program [2] which used a fixed noise strength matrix. However, for this evaluation the PF uses the Van Loan method which allows greater fidelity in modeling the discrete-time noise strength. Like the EKF and UKF implementations, the PF uses the linearized \mathbf{F} matrix in equation (8.48) to calculate \mathbf{Q}_k for this evaluation. Yet, unlike the EKF and UKF, the PF requires performing a Cholesky decomposition of \mathbf{Q}_k when propagating the uniformly weighted estimates shown pictorially in Figure 8.1 on page 166 as the final step in the PF cycle. Describing this step

mathematically, let \mathbf{R}_c be an upper-triangular square root matrix of \mathbf{Q}_k such that,

$$\mathbf{Q}_k = \mathbf{R}_c^T \mathbf{R}_c \quad (8.54)$$

Then the particles that approximate the transitional prior density appear as,

$$\mathbf{x}_{k+1}^i = \mathbf{f}(\mathbf{x}_k^i) + \mathbf{R}_c^T \mathbf{w} \quad (8.55)$$

describing the propagation from state \mathbf{x}_k to state \mathbf{x}_{k+1} according to the system dynamics and the random process noise, where $\mathbf{w} \sim \mathcal{N}(0, 1)$.

Certain situations, such as very low process noise, can result in \mathbf{Q}_k being a positive *semi-definite* matrix and not a positive *definite* matrix causing the Cholesky decomposition to fail. Therefore, to obtain a square root matrix when \mathbf{Q}_k is positive semi-definite the PF implementation in the evaluation of the different nonlinear filters uses an efficient method from the literature [111] to calculate an upper-triangular square root decomposition that satisfies equation (8.54).

This method first performs a singular value decomposition of the symmetric matrix \mathbf{Q}_k such that,

$$\mathbf{Q}_k = \mathbf{U} \boldsymbol{\sigma} \mathbf{V}^T \quad (8.56)$$

where $\boldsymbol{\sigma}$ is a diagonal matrix consisting of the eigenvalues of \mathbf{Q}_k and (\mathbf{U}, \mathbf{V}) are unitary matrices where $\mathbf{U} = \mathbf{V}$. Therefore,

$$\mathbf{Q}_k = \mathbf{V} \boldsymbol{\sigma} \mathbf{V}^T \quad (8.57)$$

$$\mathbf{Q}_k = \mathbf{V} \sqrt{\boldsymbol{\sigma}} \sqrt{\boldsymbol{\sigma}} \mathbf{V}^T \quad (8.58)$$

where $\sqrt{\boldsymbol{\sigma}} \mathbf{V}^T$ is the square root matrix of \mathbf{Q}_k . In order to make the square root matrix upper-triangular, this method next performs a QR decomposition of $\sqrt{\boldsymbol{\sigma}} \mathbf{V}^T$ to obtain the matrix \mathbf{R}_c in equation (8.54). In practice, it is helpful to take the absolute value of \mathbf{R}_c with this method to account for small imaginary components of the matrix resulting from numerics as was done in this research.

Finally, the implementation of the PF described in this chapter numerically differentiates the nonlinear dynamics model in equation (8.5). This is distinct from the implementation in Chapter V where the propagation model for the horizontal dynamics used a linear approximation (see equation (5.15)). Analysis of the linearization errors for the earlier results showed that the difference between the two implementations were insignificant. However, to represent the nonlinear dynamics with greater fidelity and increase system robustness, the PF implementation now numerically differentiates the nonlinear dynamics model for the evaluation of the different nonlinear filters. Section 8.3 discusses the methodology used for this evaluation.

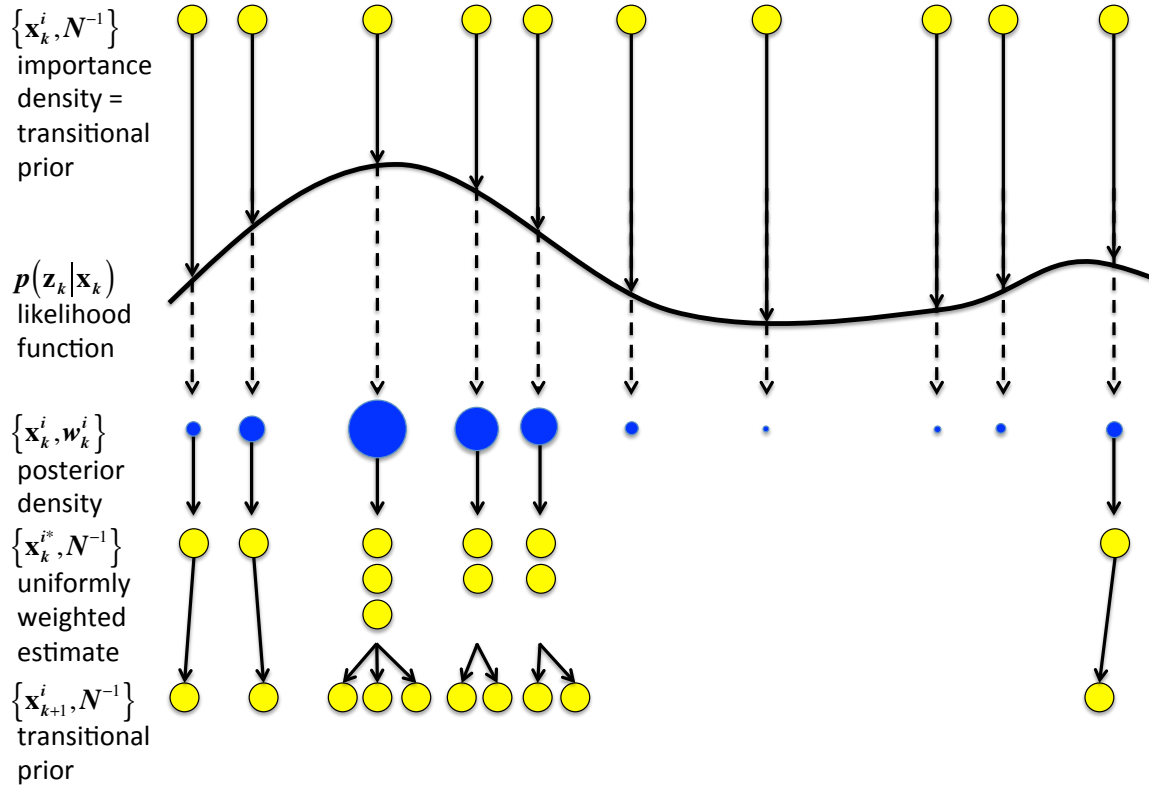


Figure 8.1: Particle Filter Algorithm. Adapted from [6, 7]

8.3 Evaluation Methodology

One of the most critical requirements for a collision avoidance system is an accurate and timely estimate of an intruder's position so as to avoid a collision. Therefore, the performance measures for this evaluation are accuracy and computational efficiency. For accuracy, the filter's performance is evaluated in two areas: (1) accuracy in estimating an intruder's current position, and (2) accuracy in predicting an intruder's future position. The estimation accuracy metric is position error, $\epsilon_p(k)$, which is calculated as,

$$\epsilon_p(k) = \frac{1}{N} \sum_{i=1}^N \sqrt{(x_{k_{\text{true}}} - \hat{x}_{k_i})^2 + (y_{k_{\text{true}}} - \hat{y}_{k_i})^2 + (z_{k_{\text{true}}} - \hat{z}_{k_i})^2} \quad (8.59)$$

where \hat{x}_{k_i} , \hat{y}_{k_i} , and \hat{z}_{k_i} are the intruder's 3D position estimates in cartesian coordinates at time k for run i . The prediction accuracy metric is turn-rate error, $\epsilon_\omega(k)$, which is calculated simply as,

$$\epsilon_\omega(k) = \frac{1}{N} \sum_{i=1}^N \sqrt{(\omega_{k_{\text{true}}} - \hat{\omega}_{k_i})^2} \quad (8.60)$$

where $\hat{\omega}_{k_i}$ is the intruder's turn-rate estimate at time k and run i . In this evaluation, $N = 1,000$ for both equations (8.59) and (8.60). Since the optimal control problem must predict the intruder's trajectory over the next receding time horizon, the $\epsilon_\omega(k)$ error metric helps quantify bounds on the future trajectory uncertainty for a maneuvering aircraft. For example, a turn-rate error of 0.5 deg/sec results in a 15° trajectory error at the end of a 30-second receding horizon prediction. With an intruder's true airspeed of 300 ft/sec, this 0.5 deg/sec turn-rate error results in an approximate 1,100 feet position error after 30 seconds. On the other hand, a turn-rate error of 6 deg/sec results in a 180° trajectory error at the end of a 30-second receding horizon prediction. This large estimation error results in a nonsensical optimal control avoidance solution since the predicted trajectory is in the opposite direction of the true trajectory.

In addition to basic modeling and filter limitations, various other uncertainty sources such as pilot intent or sensor performance limitations make having a minimum separation distance buffer around the intruder prudent to further safeguard against the possibility of a collision. This minimum separation buffer as defined in [2] is 820 feet (250 meters).

Therefore, this evaluation also highlights position errors greater than 820 feet to indicate conditions of marginal performance by the filter.

Like the accuracy metrics which are quantifiable, the metric for computational efficiency in this evaluation is CPU processing time. Because the algorithms in this research are optimized more for robust post-processing analysis than computational efficiency, the efficiency assessment is primarily a relative comparison of the filter's CPU processing times. Since the filters share a common implementation strategy for post-processing analysis, this relative comparison is a fair performance assessment. Certainly, greater efficiencies are possible with more streamlined and tailored application specific software.

Although not a quantifiable metric, another performance measure this evaluation examines is the filter's robustness to measurement geometry since certain relative ownship-intruder geometries can reduce observability and lead to poor filter performance. Because the impacts of measurement geometry contributes to the results in this evaluation, the next section first analyzes the observability grammian for a simple 2D standard-rate turn scenario because rank deficiencies in the grammian resulting from geometry will indicate scenarios where the filter's performance will be degraded.

8.4 Observability Grammian

The evaluation of the observability grammian requires a stable linear system. Since the dynamics and measurement models are nonlinear functions of the states, the EKF linearizes about the nominal estimate $\hat{\mathbf{x}}$ by taking the Jacobian of the dynamics function $\mathbf{f}(\mathbf{x}(\mathbf{t}))$ and measurement function $\mathbf{h}(\mathbf{x}(\mathbf{t}))$ to calculate \mathbf{F} and \mathbf{H} , respectively where

$$\mathbf{F} \triangleq \left. \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\mathbf{x}}} \quad (8.61)$$

$$\mathbf{H} \triangleq \left. \frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\mathbf{x}}} \quad (8.62)$$

In order to calculate the observability grammian the linearization is now performed about the true state instead of the nominal state estimate. Thus, given the linear system,

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{F}\mathbf{x}(t) + \mathbf{G}\mathbf{w}(t) \\ \mathbf{z}(t) &= \mathbf{H}\mathbf{x}(t)\end{aligned}\tag{8.63}$$

the observability grammian is defined as [112],

$$\mathbf{L}_o = \int_0^\infty e^{\mathbf{F}^T t} \mathbf{H}^T \mathbf{H} e^{\mathbf{F} t} dt\tag{8.64}$$

which is solved using the Van Loan method as described in equations (8.24) and (8.26).

For simplicity, in this 2D observability analysis scenario the ownship remains stationary while the intruder maintains a standard rate 3 deg/sec left turn while flying at a constant speed of 300 ft/sec. With this standard-rate left (counterclockwise) turn, the intruder completes a full 360° circle every 120 seconds. The ownship's north position (y-axis) in this scenario is fixed at the center of the intruder's turn circle and the ownship's east position (x-axis) is fixed at approximately 3.3 NM west of the center of the intruder's turn circle. Figure 8.2 pictorially depicts this relative geometry where the ownship is shown in blue and the intruder in red.

For this evaluation, the state propagation and observations occur at a 1 Hz rate. The scenario lasts 240 seconds with the intruder flying two complete circles. This scenario is repeated 1,000 times for each of the three different filters. The noise corrupted observations are generated from the truth data by adding random noise from a zero-mean, Gaussian distribution with variance defined by the sensor performance as shown in equation (8.4). Further, to standardized the comparison across the filters, identical noise corrupted observations values are used for the EKF, UKF, and PF runs.

8.5 Observability Results

Based on the geometry in this scenario, the areas of reduced observability corresponded to points on the intruder's turn circle that were along the lines of tangency from the ownship's position. In this scenario, these positions corresponded to 7 and 56 seconds from the start of the scenario as seen in Figure 8.3. These areas of reduced observability

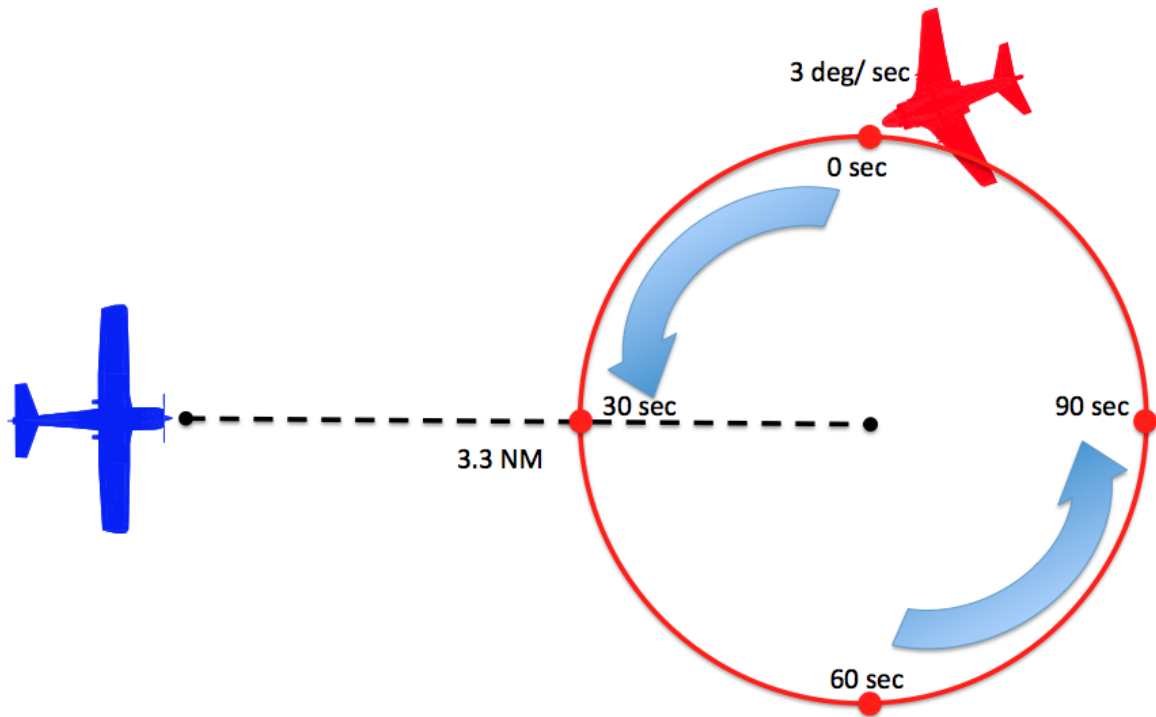


Figure 8.2: 2D Scenario for Observability Analysis

were reflected by the sharp drops in the condition number of the observability grammian in Figure 8.4a. To emphasize the repeatability of these reduced observability areas, the intruder in this scenario completed two circles which highlighted the reoccurrence of these reduced observability times at set 120 second intervals. Figure 8.4b depicts the singular values of the observability grammian where the largest singular value, σ_1 , was predominately due to the turn-rate state, ω . For this observability scenario the performance of the EKF, UKF, and PF with 1500 particles are shown in Figure 8.5 where the time-history of the mean position errors appear in panel (a) and the time-history of the maximum position errors appear in panel (b). All three filters experienced similar times where their mean and maximum position errors rose and fell as a result of measurement geometry. The affects of reduced observability especially magnified the linearization errors of the EKF as seen by the large mean of the position errors. Figure 8.6 shows the mean and maximum turn-rate errors for all three filters. Since the intruder in this scenario performed a standard-

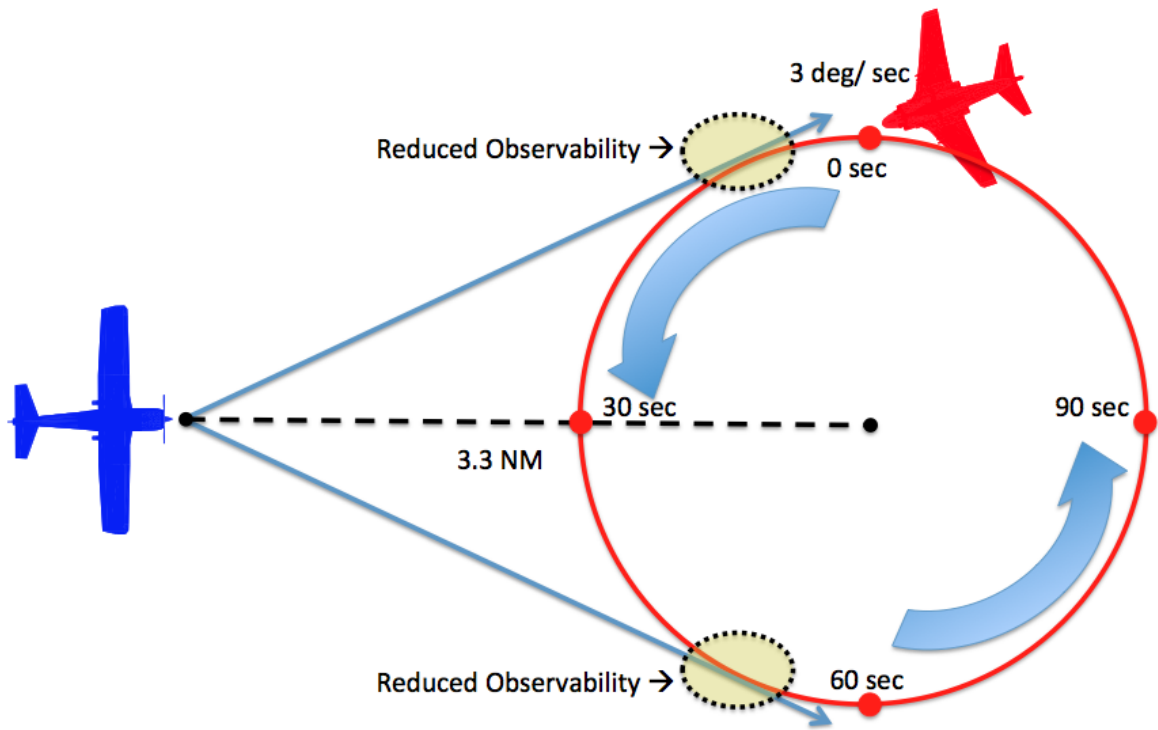


Figure 8.3: Reduced Areas of Observability Based on Geometry

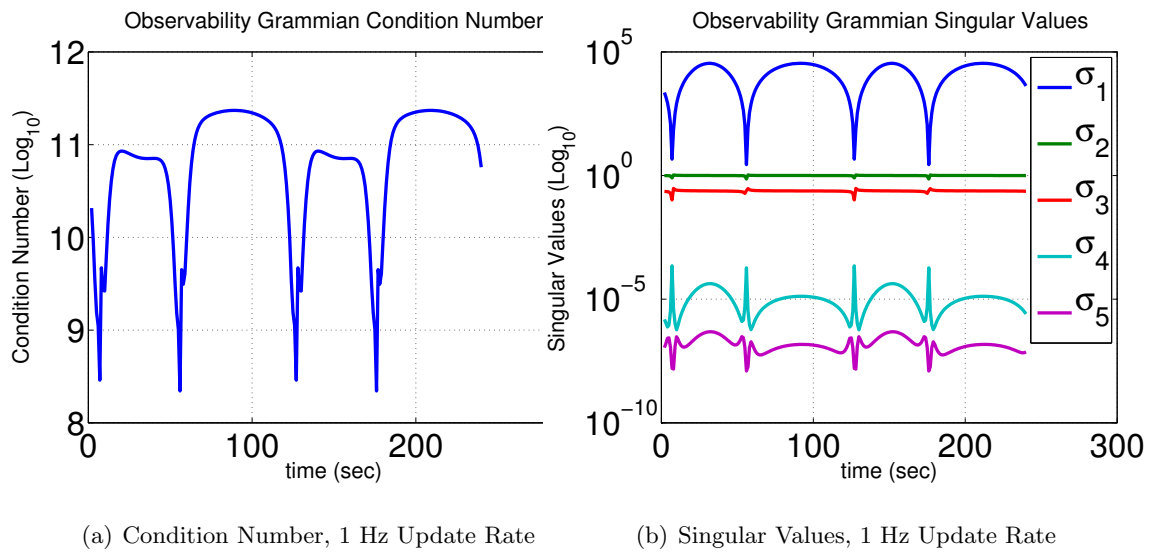


Figure 8.4: Observability Grammian Condition Number and Singular Values at 1 Hz

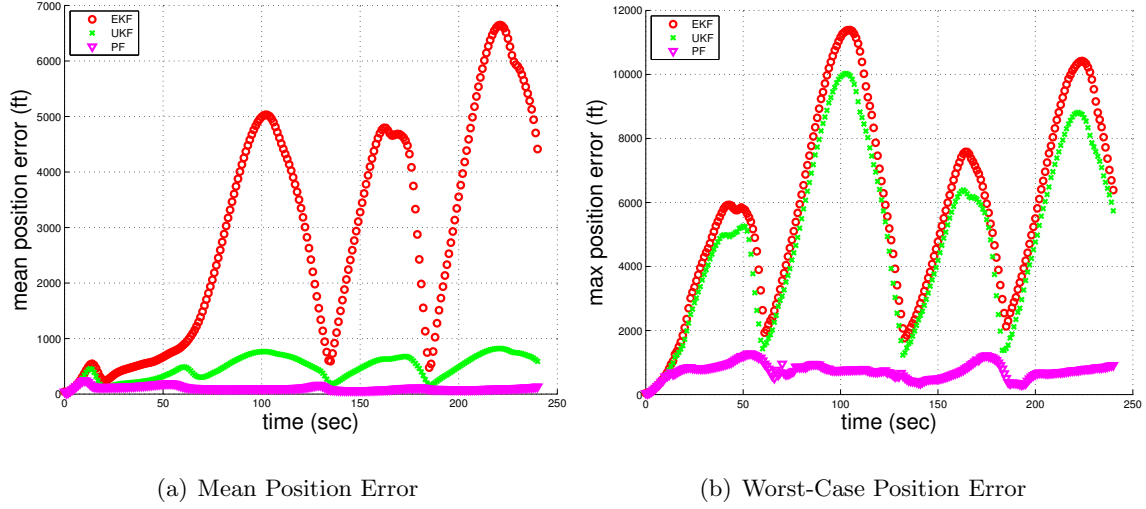


Figure 8.5: Observability Analysis Mean & Worst-Case Position Errors

rate turn, the initial mean turn-rate error for all three filters was 3 degrees. Like the

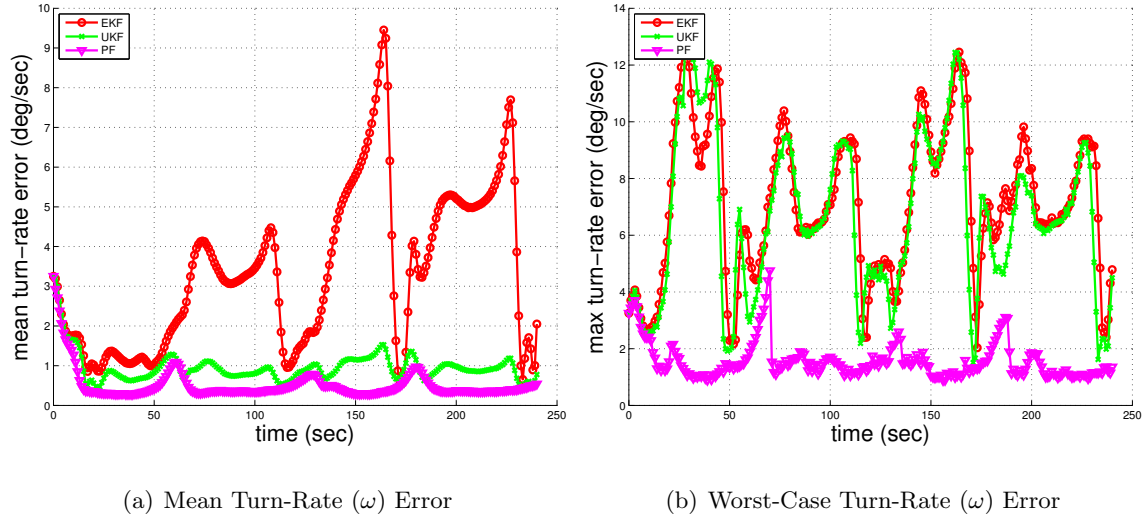


Figure 8.6: Observability Analysis Mean & Worst-Case Turn-Rate Error

position errors, the turn-rate errors for all three filters also rose and fell at similar times throughout the scenario. Yet, unlike the position errors which were a combination of errors from two different state estimates (x and y) which acted in different cardinal directions,

the turn-rate errors were directly from a single state estimate (ω), and therefore the peaks in the mean and maximum turn-rate errors corresponded closely to the times of reduced observability shown in Figure 8.3. Depending on the filter, these peak errors were significant. A faster update rate could help minimize the impacts of reduced observability resulting from adverse geometry. This research briefly explored using a faster update rate; however, a more thorough analysis of performance gains and costs resulting from a faster update rate were left as an area for further research.

In summary, this analysis showed that relative measurement geometry could lead to reduced observability and degrade filter performance. With the results of this observability analysis in mind, the next section returns to the evaluation of the filter performance starting with a description of the different evaluation scenarios. The first scenario in this evaluation is similar to the scenario in the observability analysis and is the most stressing of the evaluation scenarios.

8.6 Scenario Description

The filter evaluation consists of three scenarios that last 60 seconds each. For each scenario the ownship starts at the origin and the intruder starts to the east (positive x -axis) of the ownship. Both the ownship and intruder maintain a constant speed of 300 ft/sec. The ownship flies a constant east heading whereas the intruder performs either a standard-rate, half standard-rate, or no turn. The starting geometry for Scenario One is with both aircraft initially flying towards each other and the starting geometry for Scenario Two is with both aircraft initially flying at a 90° angle to one another. In Scenarios One and Two the ownship and intruder remain co-altitude. Scenario Three is identical to Scenario One with the exception the intruder changes altitudes and descends at a rate of 1250 ft/min while maneuvering.

8.7 Nonlinear Filter Results

Each scenario is repeated 1,000 times for the different intruder turn-rates. The results for each scenario consist of time-history plots of the mean and worst-case position and turn-

rate errors. The significance of the worst-case error plots is to provide some insight to the degree to which the mean error values may have been influenced by an extreme outlier. Also, when appropriate, a time-history plot is included showing the number of times the position errors exceeded the threshold of 820 feet.

For the particle filter results, the number of particles varies depending on the intruder model. For the scenarios in this analysis 1500 particles represents for the 5-state coordinate turn model the number of particles needed to consistently achieve performance on par or better than the EKF and UKF without experiencing divergent trajectories. Therefore, the results for Scenarios One and Two show the performance of the PF using 1500, 2000, 2500, and 6000 particles. In Scenario Three, in order to estimate the intruder's vertical maneuvers, the intruder dynamics model changes from the 5-state coordinate turn model to the 8-state model shown in equation (8.5). The literature [113] recommends when increasing the number of states (L) from L_a -states to L_b -states to scale the number of particles using the following ratio:

$$\text{scale} = \left[\frac{L_b}{L_a} \right]^3 \quad (8.65)$$

where in this case $L_a = 5$ and $L_b = 8$ so that the scale equals 4. Thus, the results for Scenarios Three show the performance of the PF using $1500 \times 4 = 6000$, $2000 \times 4 = 8000$, and $2500 \times 4 = 10,000$ particles. The results of the PF with 6000 particles in Scenarios One and Two are included in order to highlight performance differences when using a large number of particles with a lower order state model. Further, including the 6000 particle performance results in these earlier scenarios allows a side-by-side comparison of performance if electing to use this number of particles for both the 5-state and 8-state models.

8.7.1 Scenario One, Case One: Standard Rate Turn.

At the start of this scenario the intruder was approximately 4 NM east (x -axis) and approximately 1 NM north (y -axis) of the ownship. The intruder maintained a standard-rate counterclockwise turn to the south and the scenario terminated with the intruder southeast of the ownship separated by a slant range of 1.37 NM. For this analysis, this separation distance represented the nominal detection range needed in order for the ownship to execute

an avoidance maneuver and still remain outside of a 2460 feet horizontal radius from the intruder.

As demonstrated in the earlier analysis, periods of reduced observability occurred when the two aircraft initially faced each other and the intruder executed a standard-rate turn that crossed the ownship's flight path. Although the times for reduced observability in this scenario were slightly different since the ownship was no longer stationary, reduced observability still impacted and degraded filter performance in this scenario.

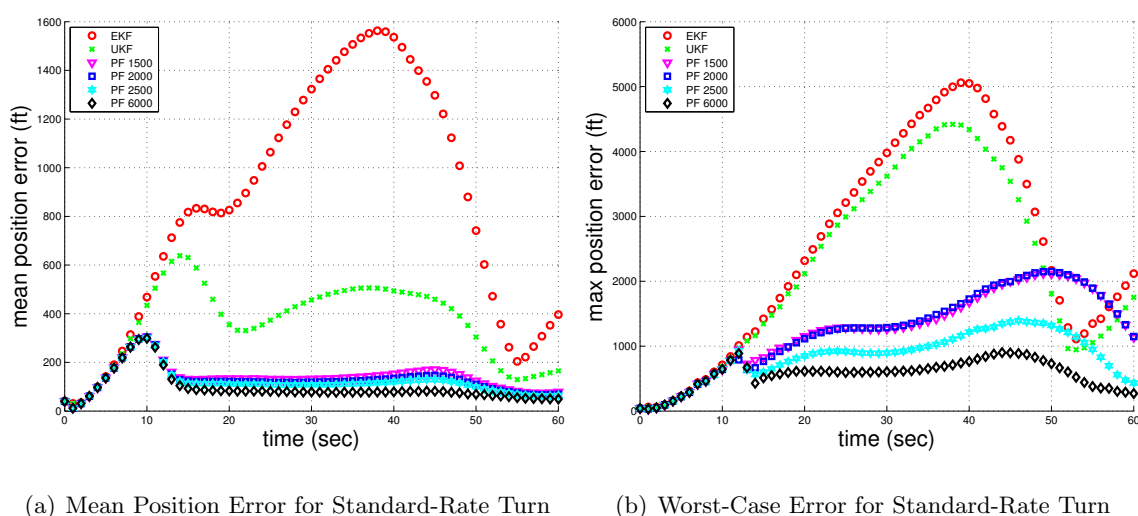


Figure 8.7: Mean & Worst-Case position Error for Standard-Rate Turn

The results for this scenario appear in Figure 8.7 where the time-history of the mean of the position errors appear in panel (a) and the time-history of the maximum position errors appear in panel (b). The performance of the EKF, UKF, and PF for different number of particles appear on this figure where the nomenclature “PF 1500” indicates the results of the particle filter

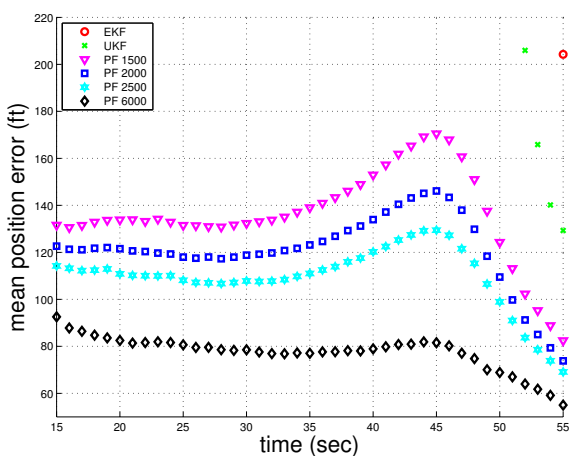


Figure 8.8: Close-In of Mean Position Errors

using 1500 particles. In this scenario, the particle filter consistently outperformed the EKF and UKF.

Figure 8.8 on the preceding page shows the position error mean results over a shorter period of time with a reduced scale to help clearly distinguish PF performance with different numbers of particles. As expected, as the number of particles increased the filter performance likewise increased. Even the worst-case errors with 2500 and 6000 particles were consistently lower than the EKF and UKF worst-case errors. Figure 8.9 depicts the number of times the position error exceeded the 820 feet threshold. The EKF and UKF results are shown in panel (a) and the particle filter results in panel (b) due to the differences in scales.

In this scenario, the reduced observability coupled with the 1 Hz update rate was particularly stressing for the EKF. Nearly 60% of the 1000 EKF trajectories exceeded the 820-foot threshold and over a third of the 1000 UKF trajectories exceeded the 820-foot threshold. From panel (b), the particle filter performed much better. Similar to the EKF and UKF, the PF trajectories first exceeded the 820-foot threshold at 12 seconds into the scenario. However, even with as few as 1500 particles the number of trajectories over

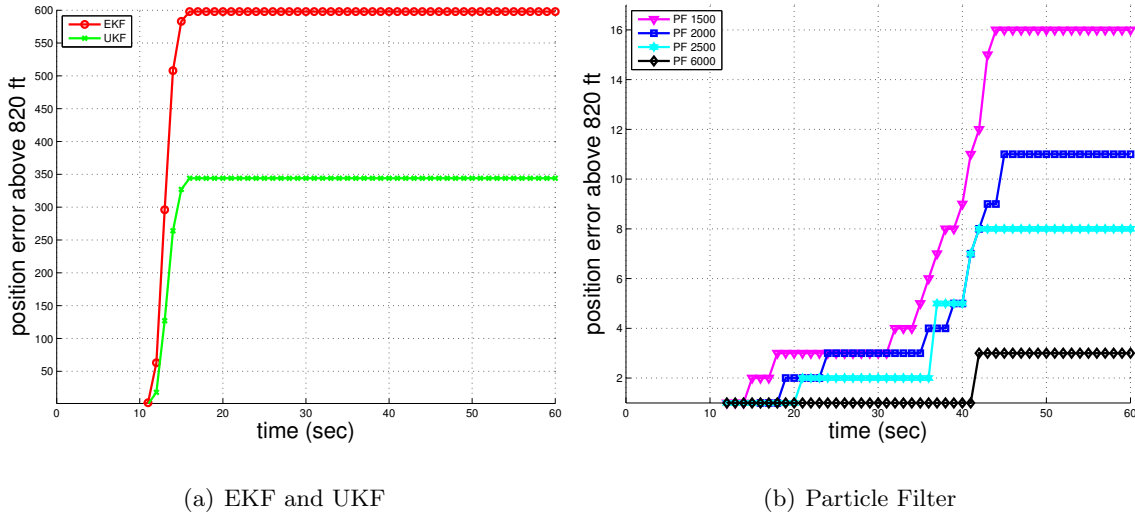


Figure 8.9: Position Errors Above 820-Foot Threshold

the 820-foot threshold was only 16 and with 6000 particles the number over the threshold decreased to three, which was nearly 200 times less than the EKF number.

Figure 8.17 shows the mean and maximum turn-rate errors for all three filters. Without a priori knowledge of the intruder's maneuvers, the initial mean estimate for the intruder's turn-rate state was zero. Since the intruder in this scenario performed a standard-rate turn, the initial turn-rate error for all three filters was approximately 3 degrees. However, the particle filter quickly drove this initial turn-rate error towards zero whereas the EKF and UKF initially decreased this error but then the error increased again at approximately 15 seconds into the scenario. Further, when the EKF and UKF worst-case turn-rate errors

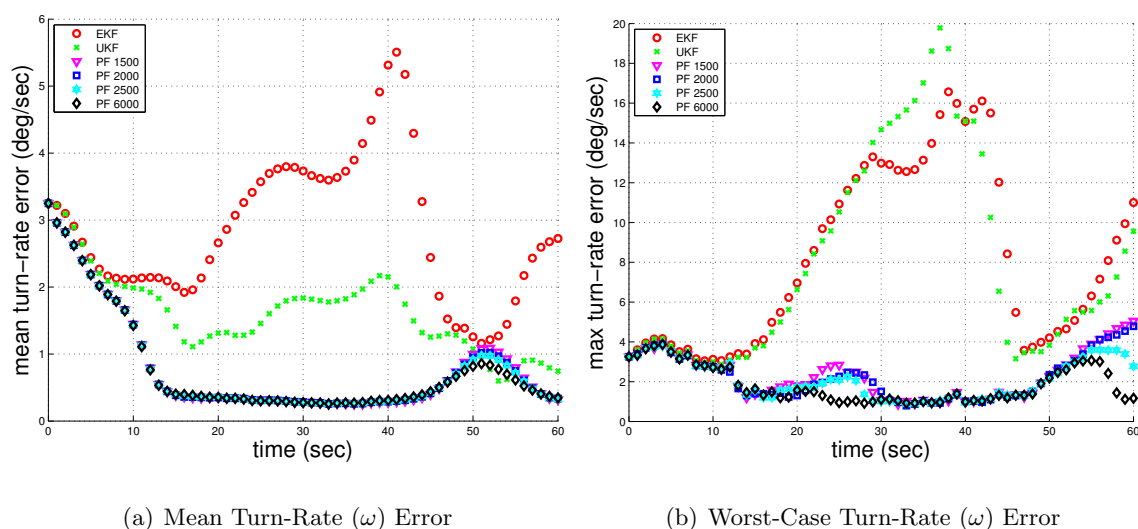


Figure 8.10: Mean & Worst-Case Turn-Rate Error for Standard-Rate Turn

peaked at 38 seconds, the PF worst-case turn-rate errors at that time were approximately an order of magnitude smaller. Unlike the position errors, the turn-rate errors did not appear to be overly sensitive to the number of particles since the mean turn-rate errors were fairly consistent from 1500 to 6000 particles. Thus, not only did the particle filter provide the best estimate and best prediction of the intruder's current and future position, the particle filter was also the most robust of the three filters to the affects of measurement geometry. However, as anticipated the increased accuracy was at a cost of computational efficiency.

Figure 8.11 shows the mean CPU time for the different filters. This mean was calculated by averaging the elapsed CPU time from the previous state estimate to the current state estimate, that is from \mathbf{x}_{k-1}^+ to \mathbf{x}_k^+ over 1000 runs with 60 measurement updates per run. Since the CPU times

were not significantly impacted by ownship-intruder geometries this calculation was not repeated until Scenario Three where the intruder state model increased from 5-states to 8-states.

As seen in Figure 8.11 as accuracy increased computational time likewise increased. The EKF was the most computationally efficient followed by the UKF and then the particle filter. The mean UKF CPU time was 33% greater than the mean EKF CPU time and the mean PF CPU time with 1500 particles was 337% greater than the mean EKF CPU time. Nonetheless, even with 6000 particles the mean CPU time was only 8.5 msec, which showed that the filter could reasonably accommodate a significantly faster update rate than 1 Hz. Additionally, for a small incremental increase in the number of particles from 1500 to 2000 to 2500, the corresponding increase in mean CPU times appeared nearly linear.

8.7.2 Scenario One, Case Two: Half Standard-Rate Turn.

Whereas the previous scenario was intentionally stressing, this current scenario was intentionally not. At the start of this scenario the intruder was approximately 5.25 NM east (x -axis) and approximately 1.89 NM north (y -axis) of the ownship. The intruder maintained a half standard-rate counterclockwise turn to the south and the scenario terminated with the intruder directly abeam the ownship separated by a slant range of 2460 feet.

The initial geometry for this scenario minimized conditions of reduced observability that degraded filter performance as seen in the previous scenario. Like the previous scenario, both aircraft initially flew towards each other; however, in this scenario the intruder only performed a half standard-rate turn and the intruder did not cross the ownship's intended

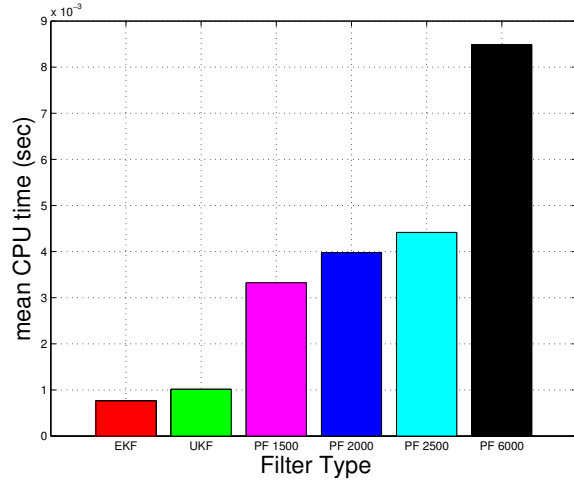
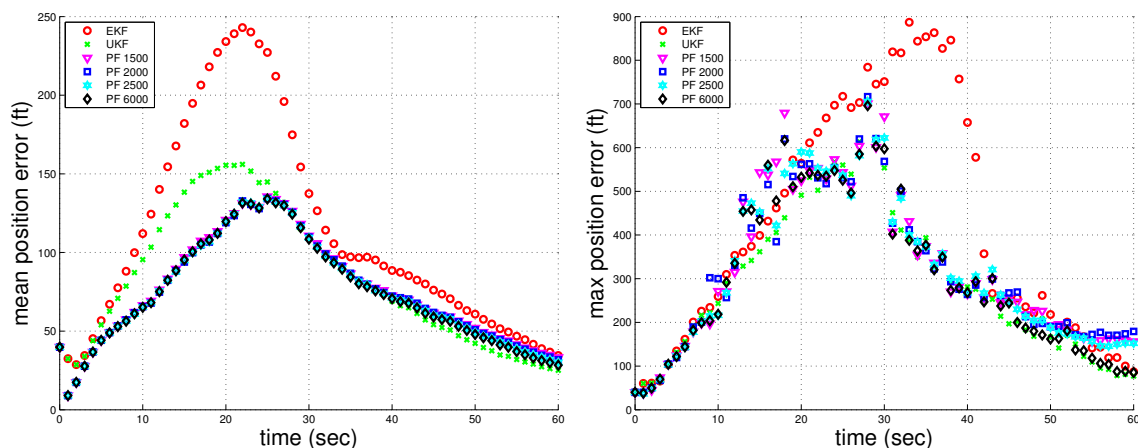


Figure 8.11: Mean CPU Time

flight path. Further, the scenario terminated with the two aircraft much closer together. This shorter separation distance coupled with the starting geometry markedly improved the performance of the EKF and UKF. The results of this scenario appear in Figure 8.12 where the mean position errors appear in panel (a) and the maximum position errors in panel (b).



(a) Mean Error for Half Standard-Rate Turn (b) Worst-Case Error for Half Standard-Rate Turn

Figure 8.12: Mean & Worst-Case Position Error for Half Standard-Rate Turn

In general, the UKF performance was on par with the performance of the particle filter. As seen in Figure 8.12, after 26 seconds the UKF mean position errors were essentially the same, and for a brief period, were even slightly better than the PF mean position errors. Also, unlike the previous scenario, the differences in the worst-case error trajectories were not as pronounced and only the EKF exceeded the 820-foot threshold and this occurred only twice.

Figure 8.13 shows the mean and maximum turn-rate errors for all three filters. Again, without a priori knowledge of the intruder's maneuvers, the initial mean estimate for the intruder's turn-rate state was zero. Since the intruder in this scenario performed a half standard-rate turn, the initial turn-rate error for all three filters was approximately 1.5 degrees. Again the PF outperformed the other filters in quickly reducing this initial error,

yet interestingly the PF turn-rate errors experienced a small peak at 25 seconds whereas the EKF and UKF errors did not. Nonetheless, after this peak the PF turn-rate errors then continued to decrease below the EKF and UKF errors by the end of the simulation. Although the EKF still had the largest worst-case turn-rate errors, the differences again were not as pronounced as seen in the previous stressing scenario.

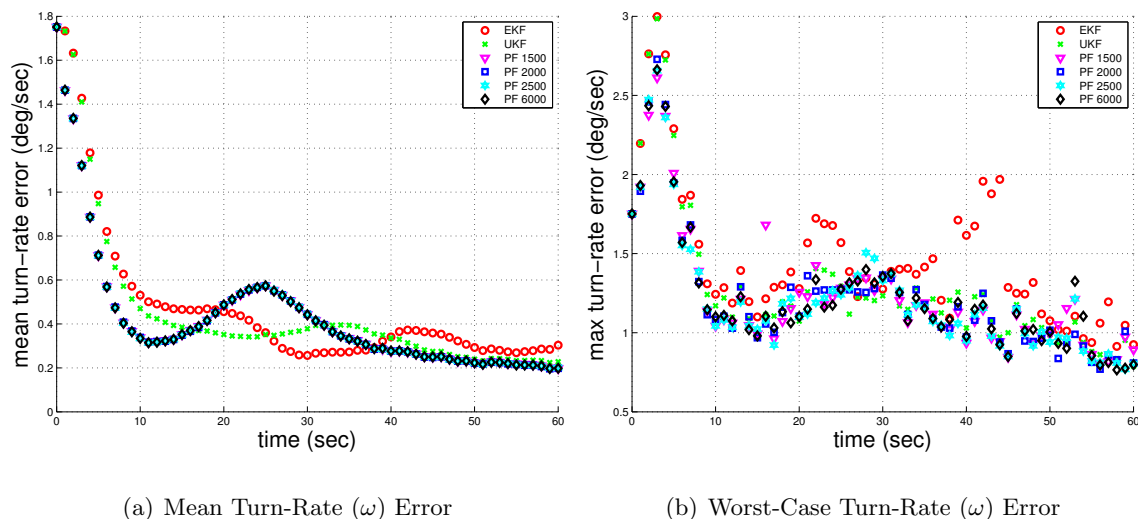
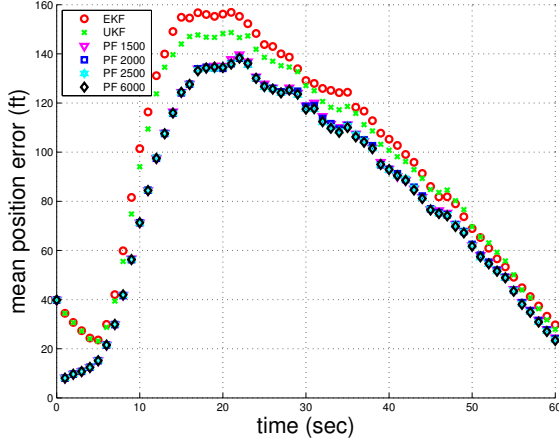


Figure 8.13: Mean & Worst-Case Turn-Rate Error for Half Standard-Rate Turn

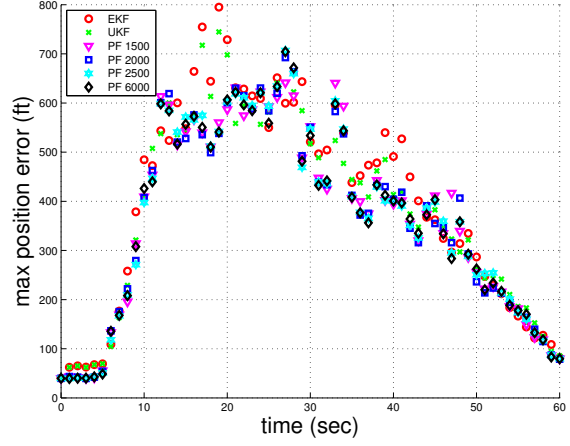
8.7.3 Scenario One, Case Three: No Turn.

Again, both aircraft flew towards each other at the start of this scenario with the intruder starting approximately 6.28 NM east (x -axis) and 0.2 NM north (y -axis) of the ownship. The intruder did not turn but maintained a west heading and the scenario terminated with the two aircraft facing each other separated by a slant range of 2460 feet.

This linear propagation scenario was not a stressing scenario for the filters. The results of this scenario appear in Figure 8.14 where the mean of the position errors appear in panel (a) and the maximum position errors in panel (b). Although the filters performed similarly in this scenario, the particle filter consistently had smaller position and turn-rate errors. For this scenario, none of the filter's position errors were over the 820-foot threshold.

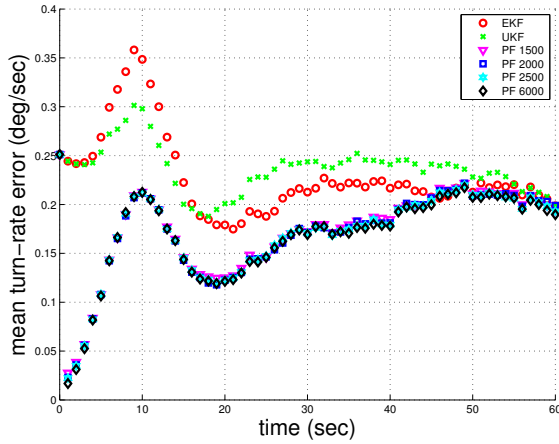


(a) Mean Position Error for No Turn

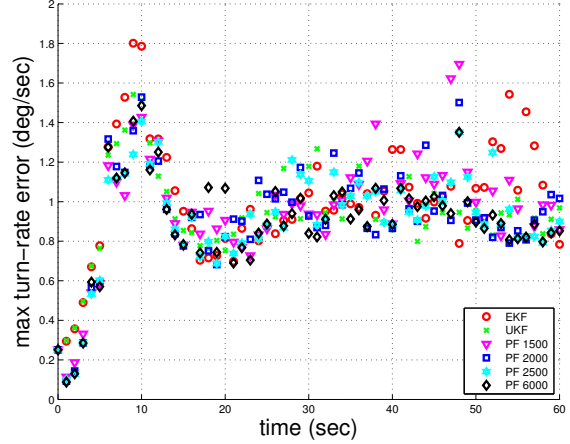


(b) Worst-Case Position Error for No Turn

Figure 8.14: Mean & Worst-Case Position Error for No Turn



(a) Mean Turn-Rate (ω) Error



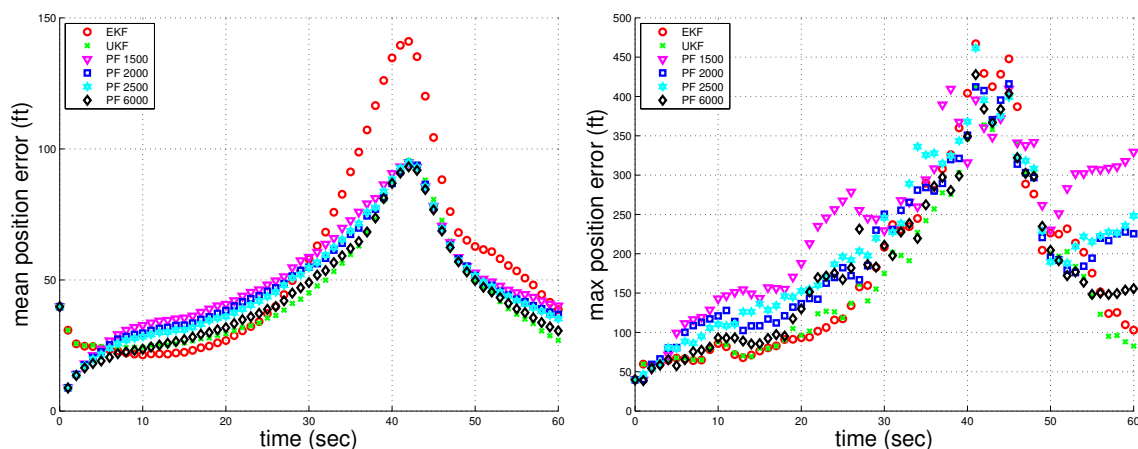
(b) Worst-Case Turn-Rate (ω) Error

Figure 8.15: Mean & Worst-Case Turn-Rate Error for No Turn

8.7.4 Scenario Two, Case One: Standard-Rate Turn.

At the start of this scenario the intruder was directly abeam and 5.25 NM east (x -axis) of the ownship. The intruder's initial heading was north (y -axis) which was at a 90° angle to the ownship's east heading. The intruder maintained a standard-rate counterclockwise turn

and the scenario terminated with the intruder again directly abeam the ownship separated by a slant range of 2460 feet. Once again, due to the initial setup geometry and the close separation distance at the termination, this was not a stressing scenario for the filters. The results of this scenario appear in Figure 8.16 where the mean position errors appear in panel (a) and the maximum position errors in panel (b). The mean EKF position errors



(a) Mean Position Error for Standard-Rate Turn

(b) Worst-Case Error for Standard-Rate Turn

Figure 8.16: Abeam Geometry: Mean & Worst-Case Position Error for Standard-Rate Turn

were again larger than the UKF and PF errors with the worst-case trajectory errors fairly similar for the filters. None of the filter's position errors were over the 820-foot threshold. Figure 8.17 shows the mean and maximum turn-rate errors for all three filters.

8.7.5 Scenario Two, Case Three: No-Turn.

At the start of this scenario the intruder was 3.37 NM east (x -axis) and 2.96 NM south (y -axis) of the ownship. The intruder maintained a north heading and did not turn. The scenario terminated with the intruder directly abeam the ownship separated by 2460 feet. The results of this scenario appear in Figure 8.18 where the mean of the position errors appear in panel (a) and the maximum position errors in panel (b). Figure 8.19 shows the mean and maximum turn-rate errors for all three filters. The results were consistent with the previous results where the PF outperformed the other filters.

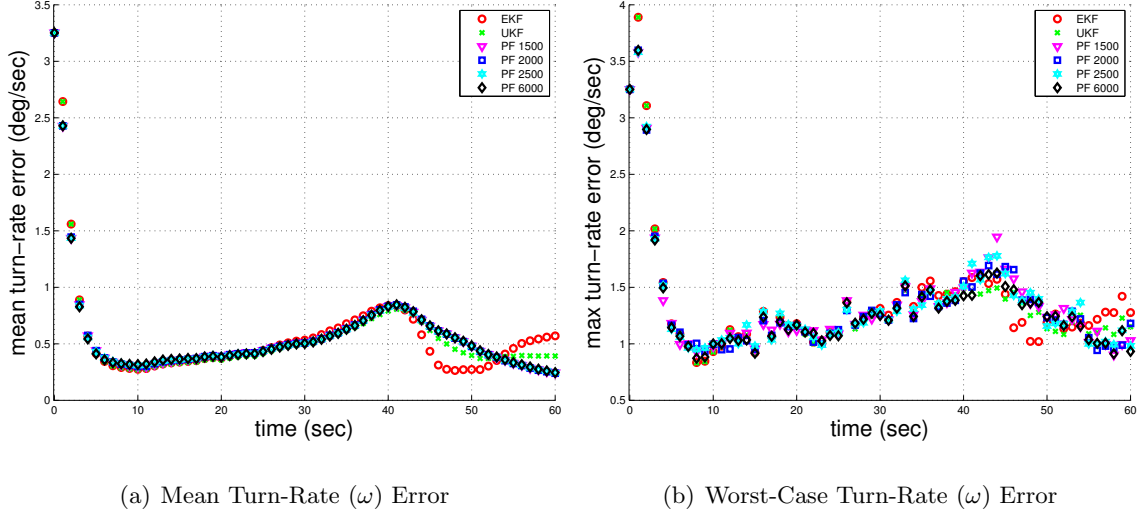


Figure 8.17: Abeam: Mean & Worst-Case Turn-Rate Error for Standard-Rate Turn

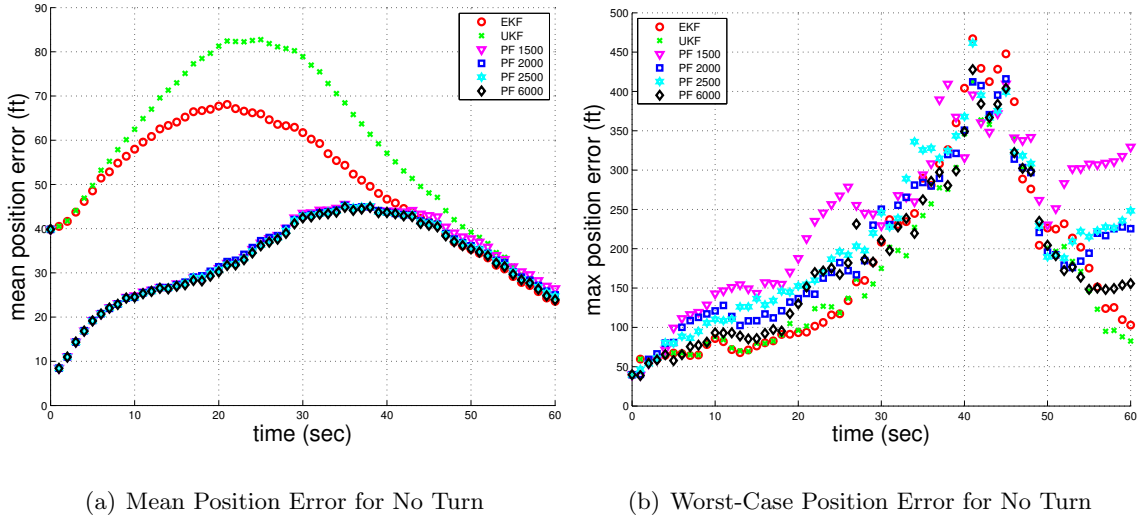
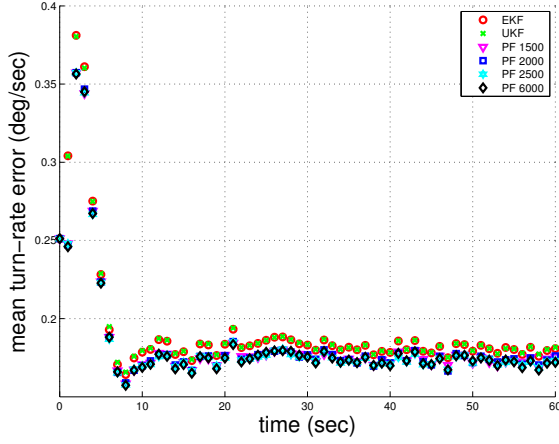


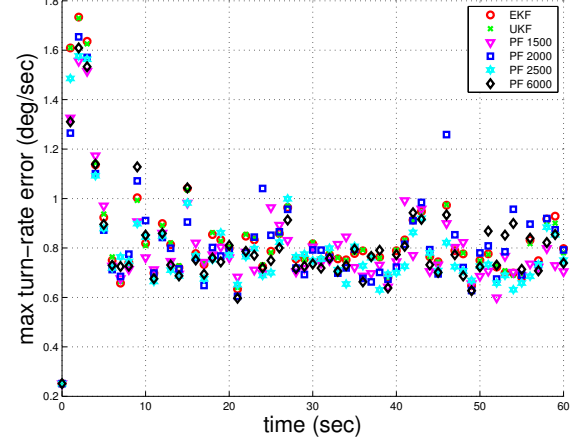
Figure 8.18: Abeam Geometry: Mean & Worst-Case Position Error for No Turn

8.7.6 Scenario Three, Case One: Standard-Rate Turn.

This scenario repeated the stressing case in Scenario One but the intruder now performed a 1250 ft/min descent rate in addition to a standard-rate turn. The results of this scenario appear in Figure 8.20. As anticipated, the altitude change by the intruder improved observability, which in turn improved the performance by the EKF and UKF;

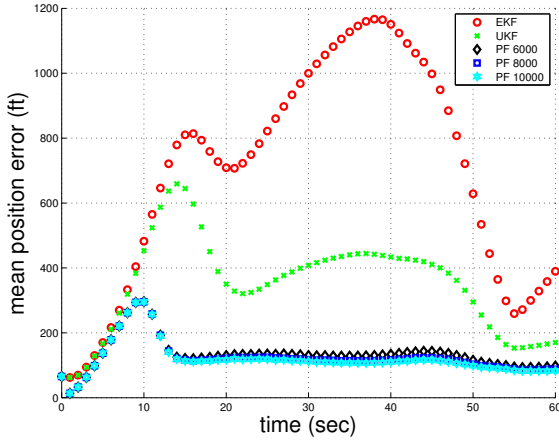


(a) Mean Turn-Rate (ω) Error

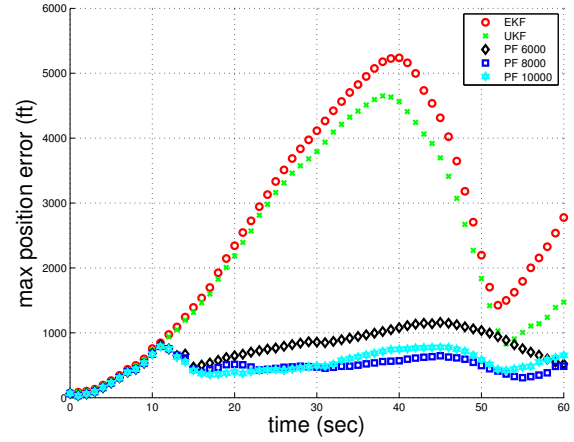


(b) Worst-Case Turn-Rate (ω) Error

Figure 8.19: Abeam: Mean & Worst-Case Turn-Rate Error for No Turn



(a) Mean Position Error for Standard-Rate Turn



(b) Worst-Case Error for Standard-Rate Turn

Figure 8.20: Mean & Worst-Case Position Error for Standard-Rate Turn

nonetheless, the particle filter was still more robust and still outperformed the EKF and UKF for both estimation accuracy and prediction accuracy in this scenario. The number of EKF and UKF trajectories that exceeded the 820-foot threshold mirrored the results in panel (a) of Figure 8.9. For the EKF, 582 of the 1000 trajectories exceeded this threshold

and for the UKF 332 of the 1000 exceeded the threshold. However, the particle filter with 6000 particles exceeded the 820-foot threshold only three times. The particle filter with 8000 and 10,000 particles did not exceed the 820-foot threshold. Figure 8.21 shows the mean and maximum turn-rate errors for all three filters. Again, the particle filter outperformed the other filters. Since the performance followed a similar trend as the previous results, the figures for the remaining scenarios in this filter evaluation are in the Appendix B to this document.

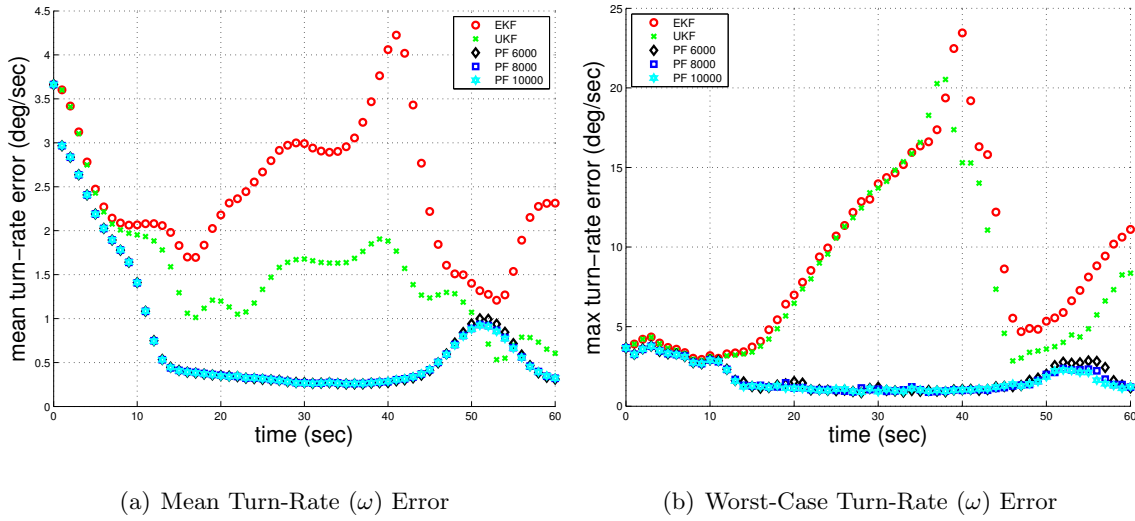


Figure 8.21: Mean & Worst-Case Turn-Rate Error for Standard-Rate Turn

Figure 8.22 shows the mean CPU times for the 8-state model. The increased number of particles caused a noticeable increase in the mean CPU times. The EKF was again the most computationally efficient followed closely by the UKF and then the particle filter. The mean UKF CPU time was 29% greater than the mean EKF CPU time and the mean PF CPU time with 6000 particles was 1,512% CPU time with 6000 particles was 1,512%

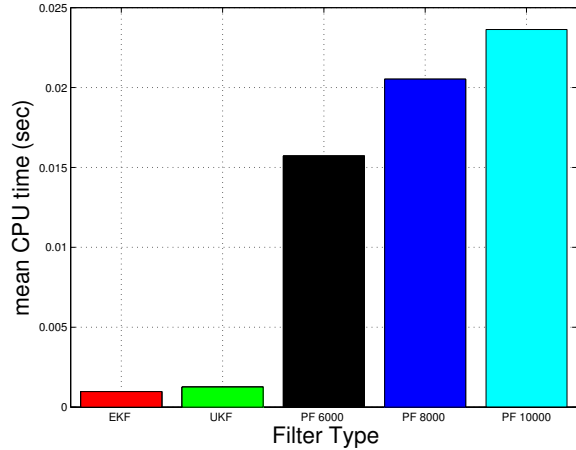


Figure 8.22: Mean CPU Time

greater than the mean EKF CPU time.

Nonetheless, the mean CPU times were only 15.7 msec for 6000 particles and 23.6 msec for 10,000 particles. Even with these large number of particles, these mean CPU times showed the PF should be able to reasonably accommodate a real-time implementation using a faster update rate than 1 Hz. With any real-time particle filter implementation a fundamental consideration is Rao-Blackwell marginalization.

In their seminal work on particle filters, [44] stressed that marginalization “is of outmost importance for high-performance real-time applications.” In marginalization the states that are linear or nearly linear are estimated using the Kalman Filter if linear or the more efficient EKF (or UKF) if nearly linear. This allows the particle filter to operate with a lower dimensionality to improve computational efficiency. In the current intruder observation and propagation models, the elevation (vertical) measurement is nonlinear, but the 3-state vertical propagation model is linear, and thus an ideal candidate for marginalization to improve overall filter efficiency. Implementation of a Rao-Blackwell marginalization scheme with an EKF or UKF for the vertical observation model and a Kalman filter for the vertical dynamics model is an area for future research. Finally, because of the inherently stochastic and non-deterministic nature, certification by the FAA of a particle filter in a safety-of-flight system for use in the NAS remains an ongoing challenge [52].

Despite these challenges of efficiency and certification, overall the particle filter has the best accuracy performance and is the most robust to adverse ownship-intruder measurement geometries. Although not as computationally efficient as the EKF and UKF, the CPU processing times for the particle filter are still viable for real-time implementation. Further, there are multiple methods to improve efficiency such as application specific hardware and software as well as taking advantage of Rao-Blackwell marginalization especially for the vertical dimension of the intruder model. These are all areas for future research. Nonetheless, for the work herein since the PF provides the most accurate and robust performance and still has reasonable computational times, the remaining analysis in this document utilizes estimates from the particle filter as inputs into the airborne collision

avoidance optimal control problem. The final ‘piece’ of the research is to demonstrate all the results. This requires combining the two keep-out regions, the ellipsoidal probability region around the intruder and the cylindrical region about the ownship, into a new inequality constraint formulation and then demonstrating this formulation in a multi-intruder stochastic collision avoidance scenario.

IX. Overlap Criteria between Cylinder and Ellipsoid

THE PREVIOUS sections of this research demonstrated methods to: (1) keep the ownship away from an ellipsoidal probability region around an intruder aircraft, and (2) keep the intruder away from a cylindrical keep-out region around the ownship. Combining these two keep-out regions in a manner that is implementable inside of the NLP is an extremely challenging problem. Complicating this challenge is the necessity of constructing a smooth everywhere differentiable function that models the gradient of the closest point of approach of these surfaces as they move relative to one another at different aspect angles. The literature contains substantial citations on algorithms to determine the overlap of hard ellipsoids in modeling molecular fluids [114] and an analytic solution exists for this problem in two dimensions [115]; however, an analytical solution does not exist [115] for finding the three dimensional distance of closest approach between two ellipsoids of arbitrary orientation or for the more complex problem of finding the distance of closest approach between a cylinder of finite height and an ellipsoid of arbitrary orientation.

9.1 Background on Cylinder-Ellipsoid Constraint Development

The authors in [114] derived an algorithm for determining if two ellipsoids, A and B , overlapped. This algorithm used the quadratic properties of an ellipsoid to construct an inside-outside functional, F_{AB} , such that [114]:

$$F_{AB}(A, B, \lambda) = \left\{ \begin{array}{l} < 1 \\ = 1 \\ > 1 \end{array} \right\} \text{ if } A \text{ and } B \left\{ \begin{array}{l} \text{overlapped} \\ \text{were externally tangent} \\ \text{did not overlap} \end{array} \right. \quad (9.1)$$

where the inputs A and B to this functional represented the principal axes, rotation angle, and centerpoint that characterized the respective ellipsoid in quadratic form and the parameter λ represented a scaled distance between the two ellipsoids that varied between $[0, 1]$. Although convenient, the downside to this inside-outside functional was that it

required an iterative search to identify the optimal value to use for the input parameter λ . Nonetheless, the research herein explored implementing this functional using a fixed (non-optimal) value for λ of 0.5 but found a significant limitation was that this approach required approximating the keep-out cylinder as an ellipsoid which resulted in either a grossly over conservative approximation or a extremely under conservative approximation since the radius (r) of the cylinder was three times the height (h) of the cylinder.

The next approach explored herein was to analytically determine the uppermost and lowermost points on the surface of an ellipsoidal probability region to use as an efficient check for feasibility. The mathematical development for this approach appears in Appendix C to this document. In this approach, if the lowest point on the ellipsoid surface was above the top of the cylinder or the highest point on the ellipsoid surface was below the bottom of the cylinder, then the trajectory was feasible. If neither of these conditions were satisfied, then this approach applied a numerical minimization algorithm, similar in concept to the search required by the first approach combined with the criteria in equation (9.1), to find the (x, y) pair on the ellipsoid surface closest to the cylinder $\forall z_{\text{ellipsoid}}$ between $\pm h$. In formulating this cylinder-ellipsoid keep-out constraint, the sigmoid product method was used to implement a set of conditional logic '*if statements.*' The feasibility region for the conditional cylinder-ellipsoid keep-out constraint was described by the following set of compounded logic *OR* conditions: If the lowest altitude point of the intruder's ellipsoidal uncertainty region was 820 feet above the ownship's altitude *OR* the highest altitude point of the intruder's ellipsoidal uncertainty region was 820 feet below the ownship's altitude *OR* the closest point on the surface of the ellipsoid to the cylinder that was between ± 820 feet of the owship's altitude was greater than the radial keep-out of 2,460 feet then the solution was feasible; otherwise, the trajectory was not feasible. This inequality constraint formulation appeared algorithmically as:

```

if min ( $z_{\text{ellipsoid}}$ )  $\geq h$ 
    feasible
else if max ( $z_{\text{ellipsoid}}$ )  $\leq -h$ 
    feasible
else if ( $x^2 + y^2 \geq r$ )
    feasible
else
    infeasible
end

```

Each of the three conditional constraints in this approach were approximated using unique sigmoid functions. This approach also required a numerical minimization algorithm to determine the (x, y) pair on the ellipsoid surface closest to the cylinder $\forall z_{\text{ellipsoid}}$ between $\pm h$ in order to evaluate the inequality constraint, $x^2 + y^2 \geq r$. This research used Matlab’s ‘*fmincon*’ function to perform this numerical minimization and then used the result as an input into GPOPS III to solve the larger collision avoidance trajectory optimization problem; however, this approach proved to be inefficient and cumbersome. Further, and more significantly, combining the numerical minimization results in a manner that the conditional constraint formulation remained smooth and everywhere differentiable was problematic. As a result, this approach was not practical, and an alternative approach was needed. The new approach was to approximate the cylindrical keep-out region using a superquadric, and then sample the ellipsoid surface at select points and compute their inside-outside function values to determine an approximate separation distance.

9.2 Cylinder-Ellipsoid Constraint Development

Superquadrics, which include a family of 3D shapes such as superellipsoids [8], have been used in a variety of applications. For example, superellipses (a 2D subset of superquadrics), have been used in the design of city streets to lofting in the design of an aircraft fuselage [8]. In the early 1980’s, Barr [116] “saw the importance of

superquadric models in particular for computer graphics and for three-dimensional design since superquadric models, which compactly represent a continuum of useful forms with rounded edges, can easily be rendered and shaded and further deformed by parametric deformations” [8]. In terms of collision avoidance, superquadrics offer tremendous potential since odd-shaped probability regions can be uniquely modeled using superquadrics. This research models the cylindrical keep-out region around the ownship using a variation of a superquadric.

From [116], the standard equation for a superellipsoid appears as:

$$\left(\left(\frac{x}{a_1} \right)^{\frac{2}{\epsilon_2}} + \left(\frac{y}{a_2} \right)^{\frac{2}{\epsilon_2}} \right)^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{z}{a_3} \right)^{\frac{2}{\epsilon_1}} = 1 \quad (9.2)$$

where the constants a_1 , a_2 , a_3 set the widths and height of the superellipsoid and the parameters ϵ_1 and ϵ_2 set the shape of the cross section parallel and perpendicular to the x, y plane respectively [8]. Figure 9.1 shows the plots of these shapes as the values of ϵ_1 and ϵ_2 varies from [0.1, 1, 2] and the values of the constants are fixed at unity, that is, $a_1 = a_2 = a_3 = 1$.

The combination of $\epsilon_1 = 0.1$ and $\epsilon_2 = 1$ in panel (d) of Figure 9.1 is a common way to represent a cylinder using a superellipsoid in computer graphics. This rendering produces a smooth tapered radius near the top and bottom of the cylinder. However, in order to minimize the tapered radius and more accurately model a true cylinder, this research uses the following modified superellipsoid equation to formulate the inequality path constraint to model the cylindrical keep-out region around the ownship:

$$\left(\frac{x}{r} \right)^2 + \left(\frac{y}{r} \right)^2 + \left(\frac{z}{h} \right)^n = 1 \quad (9.3)$$

where r is the radius and h is the height of the cylinder and n is an even natural number larger than two. Figure 9.2 shows the cylinder approximation using equation (9.3) for different values of n where $r = 2.460$ and $h = 0.820$ (representative of an aircraft keep-out region). The new inequality path constraint formulation using equation (9.3) appears as:

$$\ln \left(\left(\frac{\Delta x}{r} \right)^2 + \left(\frac{\Delta y}{r} \right)^2 + \left(\frac{\Delta z}{h} \right)^n \right) \geq \ln 1 \quad (9.4)$$

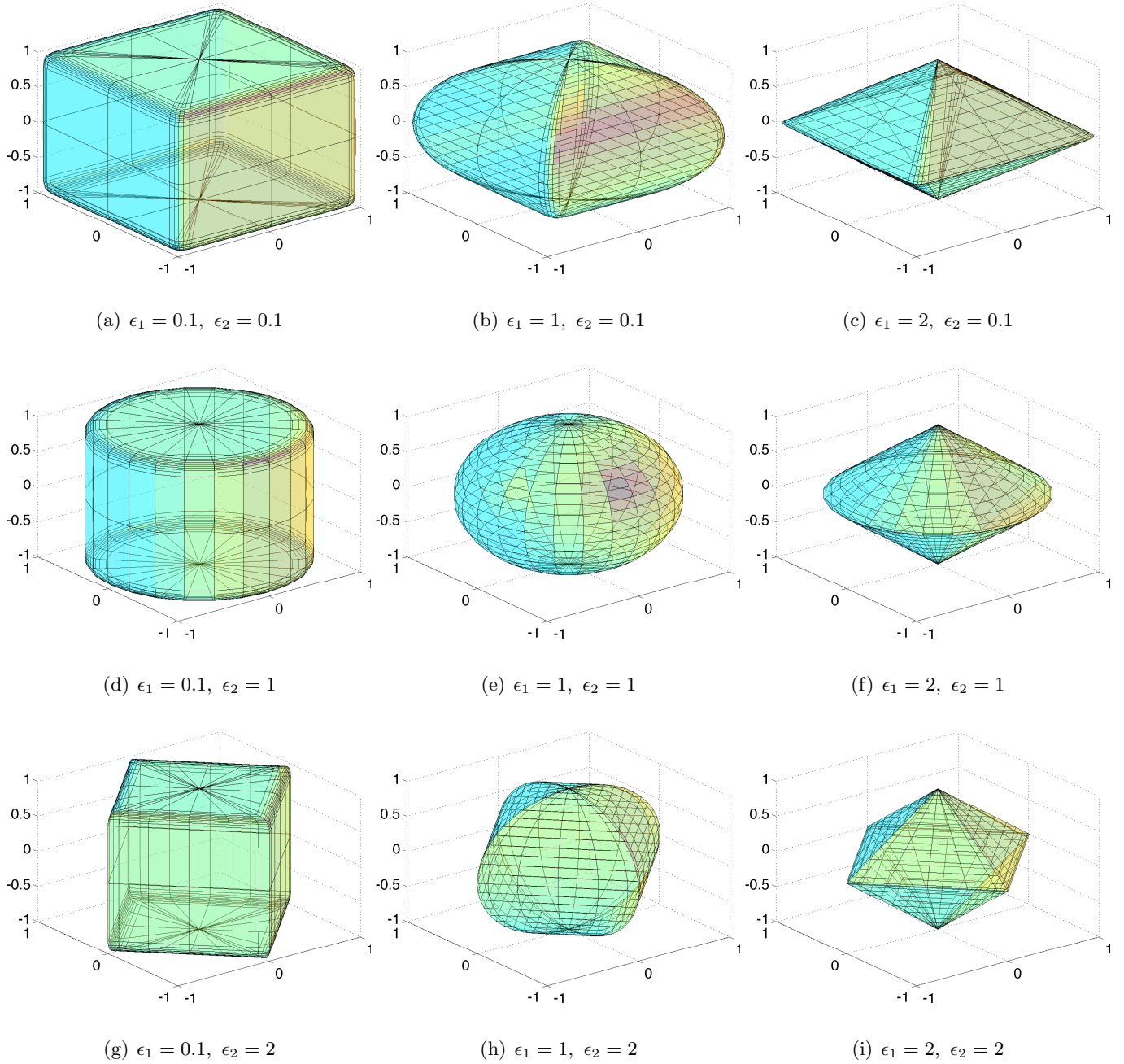


Figure 9.1: Superellipsoid for Varying Values of Exponential ϵ_1, ϵ_2 . Adapted from [8]

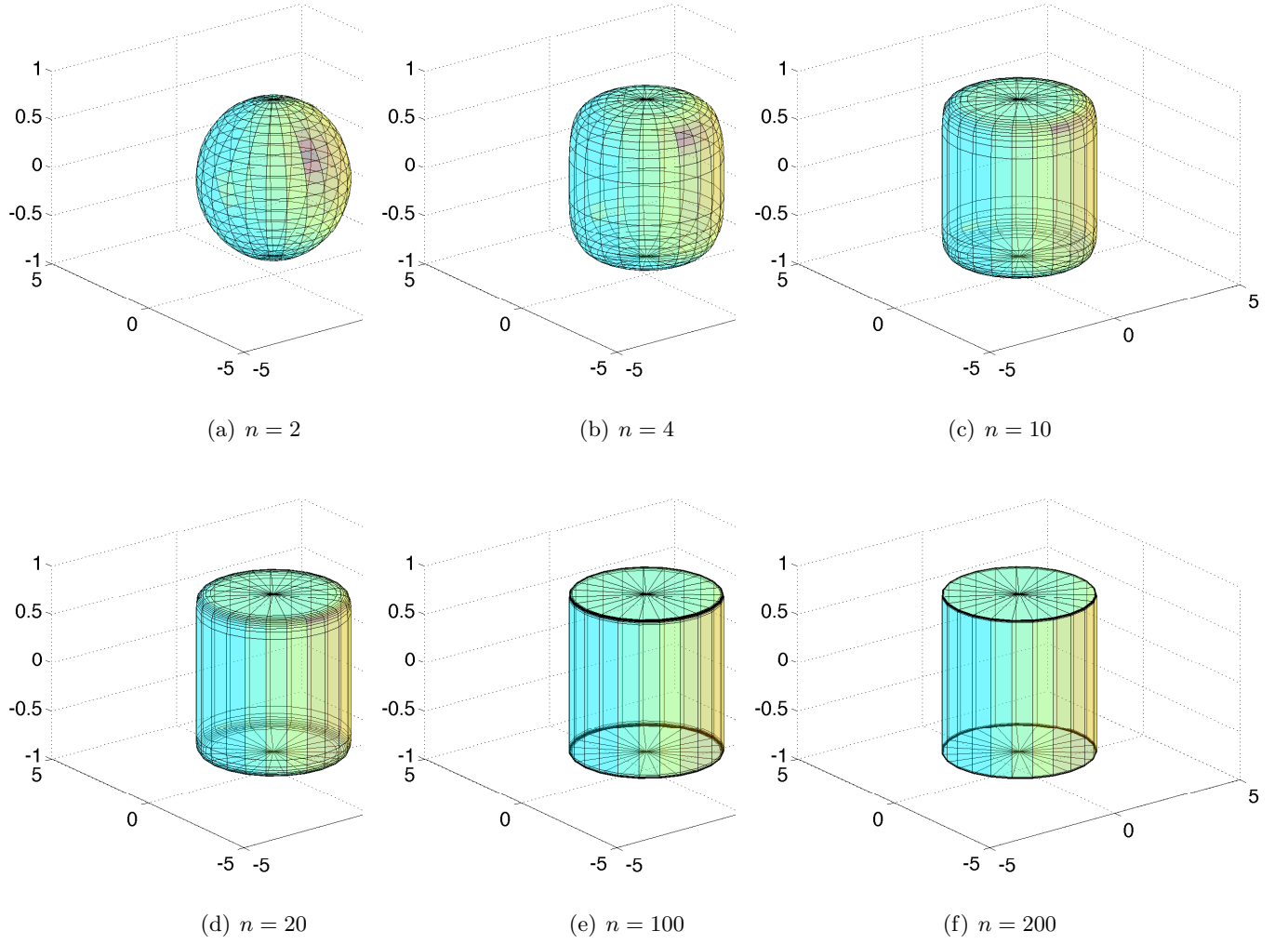


Figure 9.2: Superellipsoid Approximation of Cylinder for Varying Values of n .

and the standard form of the inequality path constraint for the optimal control problem appears as:

$$-\ln \left(\left(\frac{\Delta x}{r} \right)^2 + \left(\frac{\Delta y}{r} \right)^2 + \left(\frac{\Delta z}{h} \right)^n \right) \leq 0 \quad (9.5)$$

where Δ represents the relative difference between the intruder's outermost ellipsoid points and the ownship's position. Due to the potentially large value of the exponential power n , similar to the MAES conditional inequality constraint development earlier, the inequality constraint formulation in equation (9.5) applies the natural logarithm. Likewise, similar error bounds can be derived for equation (9.5) as developed earlier for the MAES method.

Since each intruder's probability region is approximated using an ellipsoid, this research efficiently samples points on the ellipsoid surface to evaluate the inequality path constraint in equation (9.5). This sampling is based on a parametric representation of an ellipsoid in spherical product form where the standard canonical form of an ellipsoid appears as:

$$\left(\frac{x_1}{r_1}\right)^2 + \left(\frac{x_2}{r_2}\right)^2 + \left(\frac{x_3}{r_3}\right)^2 = 1 \quad (9.6)$$

and the spherical product form of equation (9.6) appears as [116]:

$$\mathbf{x}(\eta, \nu) = \begin{bmatrix} \cos \eta \\ r_3 \sin \eta \end{bmatrix} \otimes \begin{bmatrix} r_1 \cos \nu \\ r_2 \sin \nu \end{bmatrix} = \begin{bmatrix} r_1 \cos \eta \cos \nu \\ r_2 \cos \eta \sin \nu \\ r_3 \sin \eta \end{bmatrix} \quad (9.7)$$

From equation (9.7), the ellipsoid surface is sampled at equally spaced points on a reference sphere chosen using:

$$-\frac{\pi}{2} \leq \eta \leq \frac{\pi}{2} \quad \text{and} \quad -\pi \leq \nu < \pi \quad (9.8)$$

Numerically, the $(n_e)^2$ sampled points are calculated as follows:

$$m = n_e - 1, \text{ and } \eta_i = \frac{i\left(\frac{2}{m}\right) - 1}{\pi}, \text{ where } i = \{0, 1, \dots, m\}, \text{ and } \nu_i = \frac{\eta_i}{2} \quad (9.9)$$

These points are then used in equation (9.7) to evaluate the conditional inequality path constraint in equation (9.5) for all possible combinations of (η, ν) points. In the example problem described next $n_e = 10$. This means for each collocation node the optimal control problem evaluates the inequality path constraint in equation (9.5) at $(n_e)^2$ or 100 points on the ellipsoid surface. Note that while only the closest of the 100 points is needed, it is numerically faster to check all 100 points then to find the closest of the 100 points. To help visualize the sampling density, the graphical rendering of the ellipsoidal surface for the example problem is constructed using the same sampling points evaluated in the inequality path constraint.

9.3 Cylinder-Ellipsoid Example Problem

This section demonstrates the developments in this chapter using an example problem that combines and builds on the developments from all the previous chapters. This example

problem demonstrates an optimal collision avoidance scenario in a multi-intruder stochastic environment where the inequality path constraint formulation prevents a cylindrical keep-out region around the ownship from intersecting with time-varying ellipsoidal probability regions around the intruders while minimizing an overall weighted path deviation cost functional. Algorithmically, the scenario flows as follows:

```

for each measurement update time
    estimate intruders' state vector
        use particle filter (PF 6000) described in Section 8.2.4
        use observation model described in equations (8.1) - (8.4)
        use intruder dynamics model described in equations (8.5) - (8.11)
    solve optimal control problem
        use receding horizon method described in Section 3.1.2
        use SLIMVEE algorithm described in Section 5.3.2
        use collocation node spacing described in Section 7.2.2
        use cost functional described in equation (7.1)
        use ownship dynamics model described in equations (7.5) - (7.8)
        use inequality path constraint described in equations (9.5) - (9.9)
            where  $n = 100$  and  $n_e = 10$ 
end

```

9.4 Scenario Description

The measurements in this simulation consist of simulated 1 Hz relative measurements from a radar onboard the ownship. These measurements are noise corrupted using the 3D noise model and relative measurements as described in the previous section. To perform the nonlinear estimation, this scenario uses the particle filter with 6,000 particles as developed earlier in the previous chapter. For the filter initialization, the particle filter is given 10 seconds of measurements updates to simulate a “warm-start” prior to the beginning of the scenario. This warm-up period is typical for filtering applications and mirrors true system

performance where the filter is running for a sufficient time to allow for convergence prior to the start of an engagement. For simplicity, in this initialization period the ownship remains stationary while the intruders continue to move relative to the ownship.

In this three-intruder scenario, the first two intruders are maneuvering while the third intruder maintains a constant altitude and heading. The ownship starts at the origin and maintains a constant speed of 450 ft/sec. The intruders all start to the east (positive x -axis) of the ownship. The first intruder, Intruder 1, starts 20,000 feet to the east, 8,000 feet to the south (negative y -axis), and 500 feet below (negative z -axis) the ownship. Intruder 1 flies a half-standard rate righthand turn to the north (clockwise) while maintaining a 1,250 ft/min climb and a constant speed of 400 ft/sec. The second intruder, Intruder 2, starts 25,500 feet to the east, 9,000 feet to the north, and 500 feet below the ownship. Intruder 2 flies a half-standard rate lefthand turn to the south (counterclockwise) while maintaining a 500 ft/min descent and a constant speed of 200 ft/sec. The third intruder, Intruder 3, starts at approximately 29,000 feet to the east, 6,076 feet (or 1 NM) to the north (positive y -axis), and 5,500 feet above (positive z -axis) the ownship. Intruder 3 flies a constant heading at an approximate 30° angle to the ownship's intended flight path while maintaining a constant speed of 291 ft/sec. Table 9.1 summarizes the initial conditions for the intruders in this scenario.

Table 9.1: Intruder Initial Conditions for Cylinder-Ellipsoid Scenario

Intruder	Initial Position (K ft)	Speed (ft/sec)	Turn-rate (deg/sec)	Climb (ft/min)
One	$x_0 = +20$ $y_0 = -8$ $z_0 = -0.5$	400	-1.5	+1250
Two	$x_0 = +25.5$ $y_0 = +9$ $z_0 = -0.5$	200	+1.5	-500
Three	$x_0 = +29.1$ $y_0 = +6.1$ $z_0 = +5.5$	291	0	0

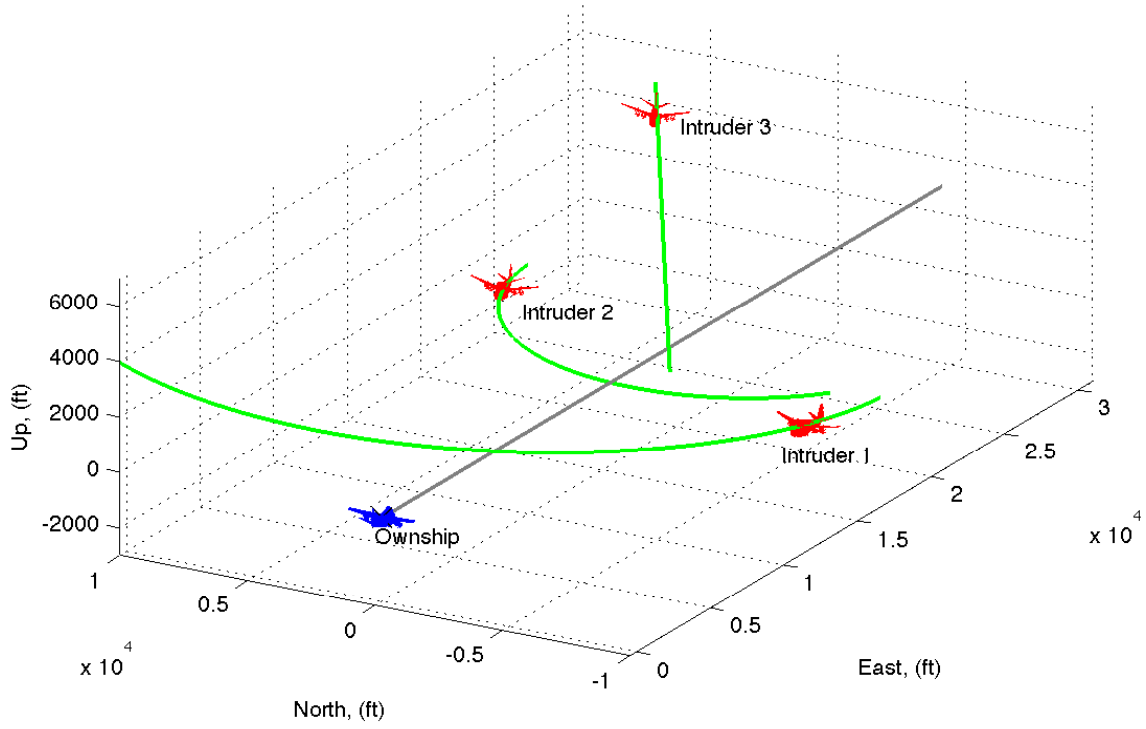


Figure 9.3: Initial Geometry for Cylinder-Ellipsoid Scenario

Figure 9.3 shows the intruder positions at the start of the scenario. The green lines in this figure represent the truth trajectory for each intruder. The small portion of green truth trajectory behind each intruder represents the aircraft movement during the filter initialization. The gray-line starting at the origin and extending east represents the ownship's intended 3D flightpath.

Like the MIAA program, the uncertainty volumes around each intruder is bounded using a lower and upper limit. The lower limit, as addressed earlier, is to account for unmodeled intruder dynamics as well other uncertainties inherent to the airborne collision avoidance problem. The upper limit in this case prevents excessive deviation maneuvers in the NAS especially since the cylindrical keep-out zone about the ownship expands nearly 5,000 feet horizontally and over 1,600 feet vertically. In this scenario the ellipsoid lower limit radius is 820 feet and the upper limit is 2,460 feet. However, because aircraft in the NAS

typically perform smooth and predictable altitude transitions, to minimize unnecessarily large uncertainty distributions in the vertical direction, the ellipsoidal probability region in the z -axis (or vertical direction) is restricted to be no more ± 820 feet of the estimated intruder altitude. The particular upper and lower limits in this scenario are merely to demonstrate the inequality path constraint formulation using bounds on the size of the probability regions. In practice, these limits can be set as desired based on the ownship maneuver capabilities and the anticipated operating environment.

The SLIMVEE algorithm developed earlier is applied in this scenario. The centroid of the ellipsoidal probability region is centered about the particle filter estimate, \mathbf{x}_k^- , which is the intruder's predicted position during the 30-second time horizon without a measurement update. The filter estimates are linearly interpolated between collocation nodes since the difference between a spherical interpolation is negligible in this scenario. Figure 9.4 shows the initial uncertainty volumes around the intruders and the cylindrical keep-out volume around the ownship at the start of the scenario.

The scenario lasts 45 seconds. For the first 15 seconds the ownship performs a measurement update every second. The measurement updates are identified in this scenario by black squares at the ownship's position and by black squares marking each intruder's estimated position. Because of the regular measurement updates, the uncertainty volumes around the intruders remain near the minimum value; however, to better demonstrate how the growth of the uncertainty volumes impact the avoidance trajectory during the 30-second horizon prediction, after the 15th second the ownship does not perform another measurement update.

9.5 Cylinder-Ellipsoid Scenario Results

Figure 9.5 shows the top and side views of the ownship and intruders at time 15 seconds. Although the slant range distance between the ownship and Intruder 1's estimated position was 1.25 NM, the ownship had already deviated from its intended flight path by 780 feet to the north (positive y -axis). This deviation was necessary to keep the large cylindrical keep-out region around the ownship from intersecting with the projected uncertainty volume

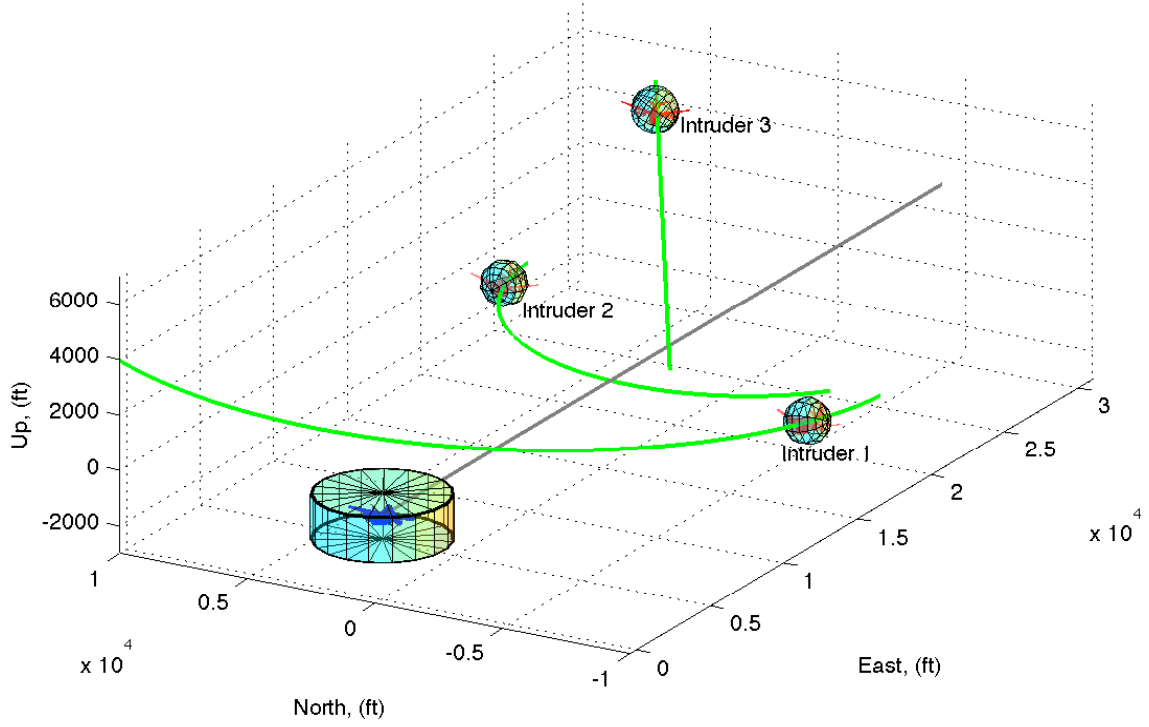
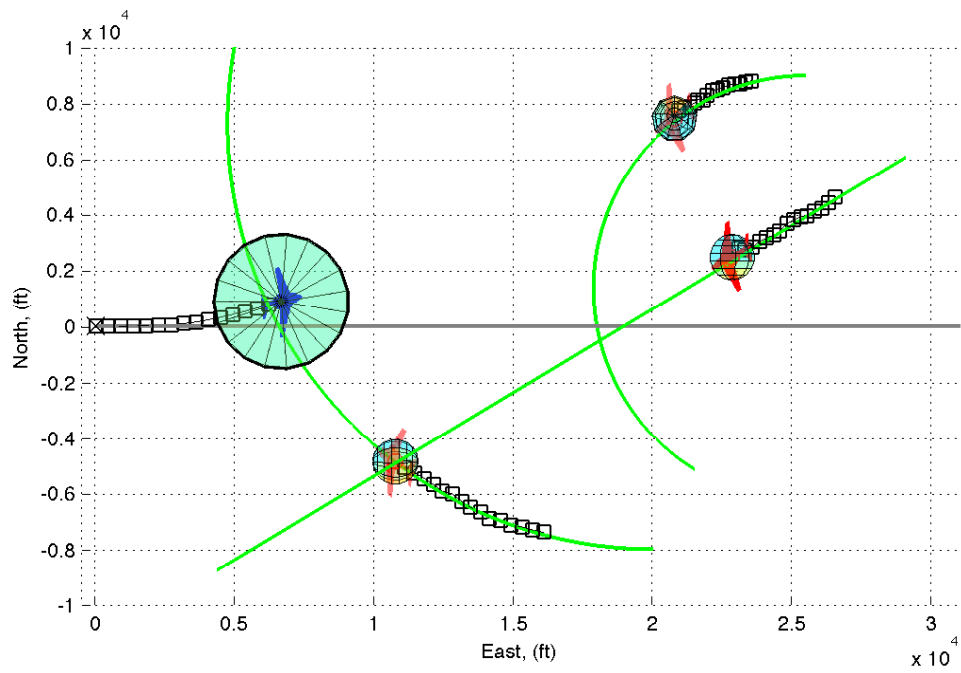


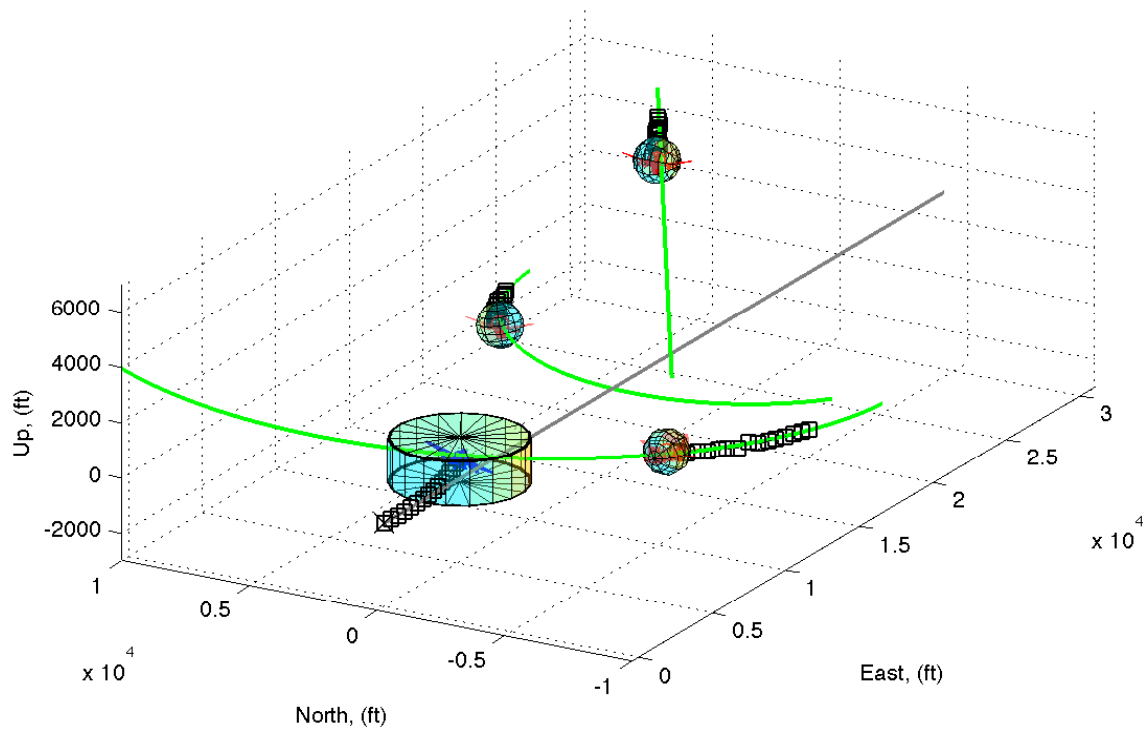
Figure 9.4: Initial Geometry and Uncertainty Volumes for Cylinder-Ellipsoid Scenario

around Intruder 1's future position. Because of the weighted cost functional which penalized vertical deviations more heavily than horizontal deviations, the ownship maintained altitude and deviated horizontally.

Figure 9.6 shows the top and side views of the ownship and intruders' position at time 25.7 seconds, approximately 10 seconds after the last measurement update. The blue circles indicate the ownship's position at each collocation node and the red circles indicate the intruder's predicted position at each node. As expected, without additional measurement updates the probability regions continued to grow. An interesting observation highlighted by Figure 9.6 was the closest point of approach between the ownship's keep-out region and Intruder 1's ellipsoidal probability region occurred after the two aircraft have passed. If the keep-out region around the ownship was not symmetrical but instead smaller behind the aircraft, then in this scenario the ownship could have returned sooner



(a) Top View

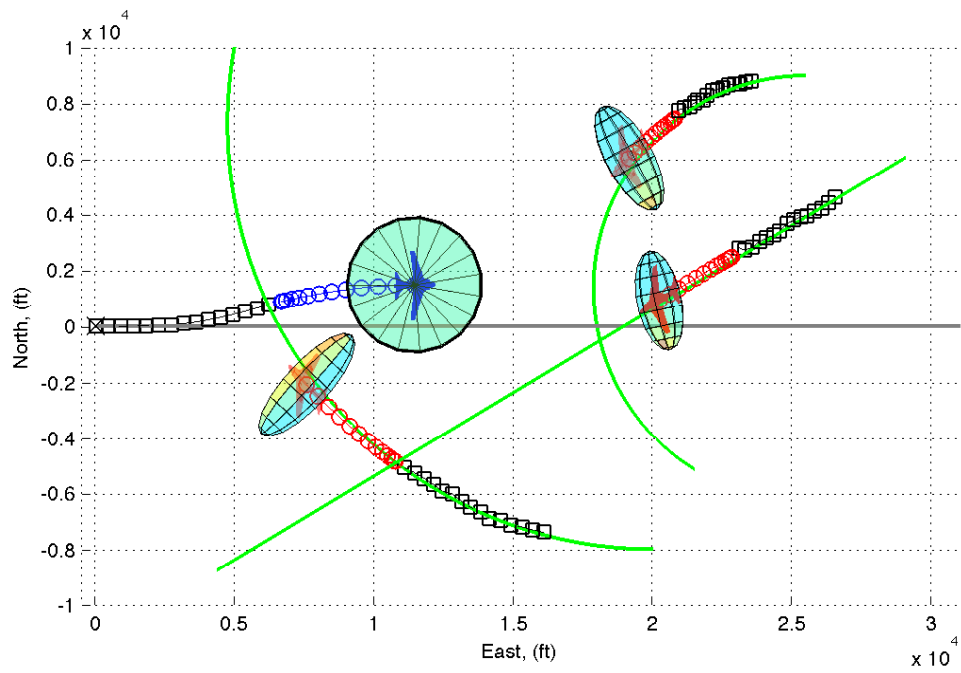


(b) Side View

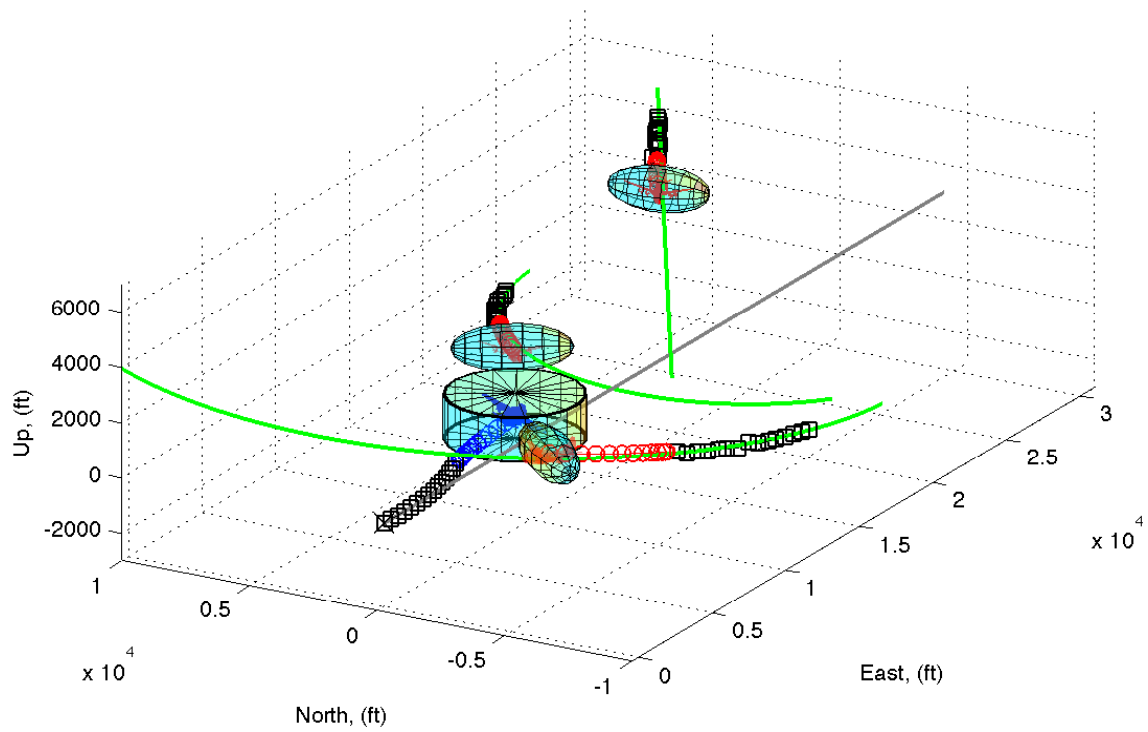
Figure 9.5: Cylinder-Ellipsoid Scenario at Time 15 Seconds

to the intended flight path. This observation validated the ongoing effort by the research community to implement a more appropriate keep-out region. Although not implemented here, the constraint formulation methods demonstrated in this research using superquadrics and sigmoids could have been applied to model a tailored keep-out region, such as one that extended further in front than in back of the ownship or a dynamic region that changed as a function of relative closure speed between the ownship and intruder. Both of these types of keep-out regions have been recently proposed earlier this year (2014) by the research community for consideration in an airborne collision avoidance application for the NAS [18, 19].

Figure 9.7 shows the position of the ownship and intruders along with their probability regions at the end of the 30-second time horizon. As seen earlier, Intruder 1's climbing half-standard rate turn caused the ownship to deviate to the north. This initial avoidance maneuver placed the ownship closer to Intruder 2 who performed a half-standard rate turn directly towards the ownship's new flight path. As a result, the ownship climbed to a maximum altitude of 635 feet and passed over the top of Intruder 2's ellipsoidal probability region. Like the previous figure, this avoidance maneuver highlighted an additional concern with the large keep-out region about the ownship. Whereas Figure 9.6 showed a concern with the large keep-out region in the horizontal direction behind the ownship, Figure 9.7 highlighted a similar concern in the vertical direction below the ownship. In this scenario, Intruder 2 started 500 feet below the ownship's initial altitude and performed a shallow 500 ft/min descent. When the ownship started the climb to pass over the top of Intruder 2's ellipsoidal probability region, the particle filter had already correctly estimated that the intruder was approximately 800 feet below the ownship in a steady descent. However, the climb by the ownship was necessary to prevent the large keep-out region that extended vertically 820 feet below the ownship's position from intersecting with the top of Intruder 2's probability region. This scenario further underscored the need for an adaptable keep-out region. For instance, in this scenario since Intruder 2 was in a steady descent, an adaptive keep-out region could have decreased the amount of required separation below the ownship



(a) Top View



(b) Side View

Figure 9.6: Cylinder-Ellipsoid Scenario at Time 25.7 Seconds

since the intruder’s velocity vector was in a direction away from the ownship. Nevertheless, given the current cylindrical keep-out region the optimization algorithm correctly calculated a feasible trajectory that satisfied the conditional inequality path constraint. The maximum horizontal deviation in this scenario was 1,484 feet and the maximum vertical deviation was 635 feet. As seen in Figure 9.7 due to the turn-rate maneuver time constant, τ_ω , the predicted positions (red circles) for both turning intruders at the end of the 30-second time horizon were further away from the green truth trajectories. This occurred because the intruders’ predicted turn-rate by design transitioned towards zero in the absence of regular measurement updates.

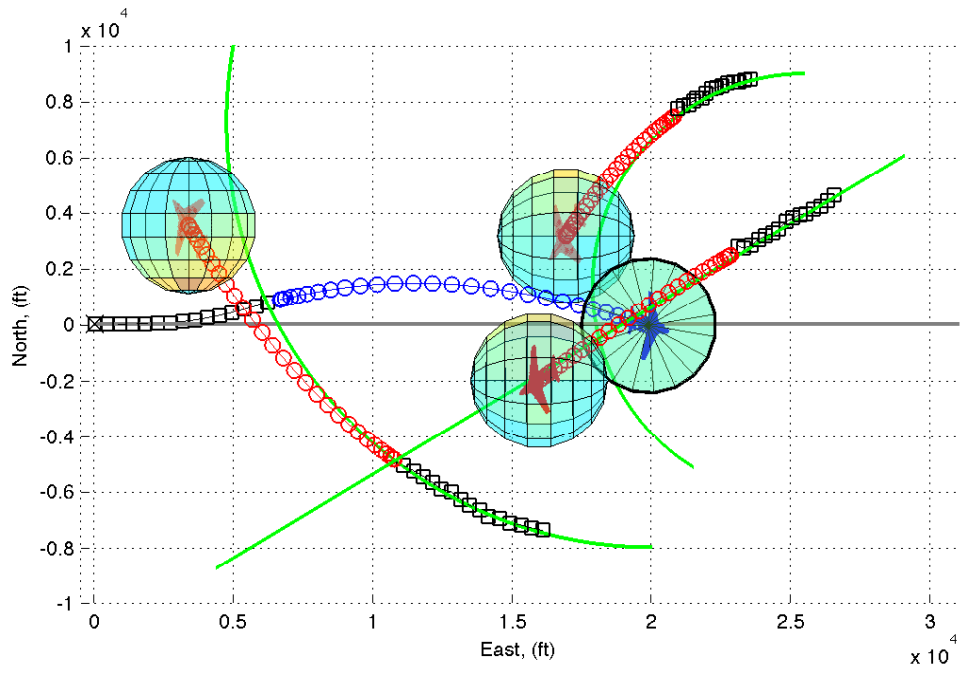
The NLP solver was SNOPT, which converged much quicker than IPOPT in this particular scenario. However, the NLP times still did not support real-time operations. Table 9.2 shows the average, the shortest, and the longest times for the NLP to calculate a solution for the 30-second time horizons. Although Intruder 3 was not an active constraint in this scenario, the algorithm still evaluated Intruder 3’s estimated position at each collocation node which affected the NLP execution times.

Table 9.2: NLP Execution Times for Cylinder-Ellipsoid Scenario

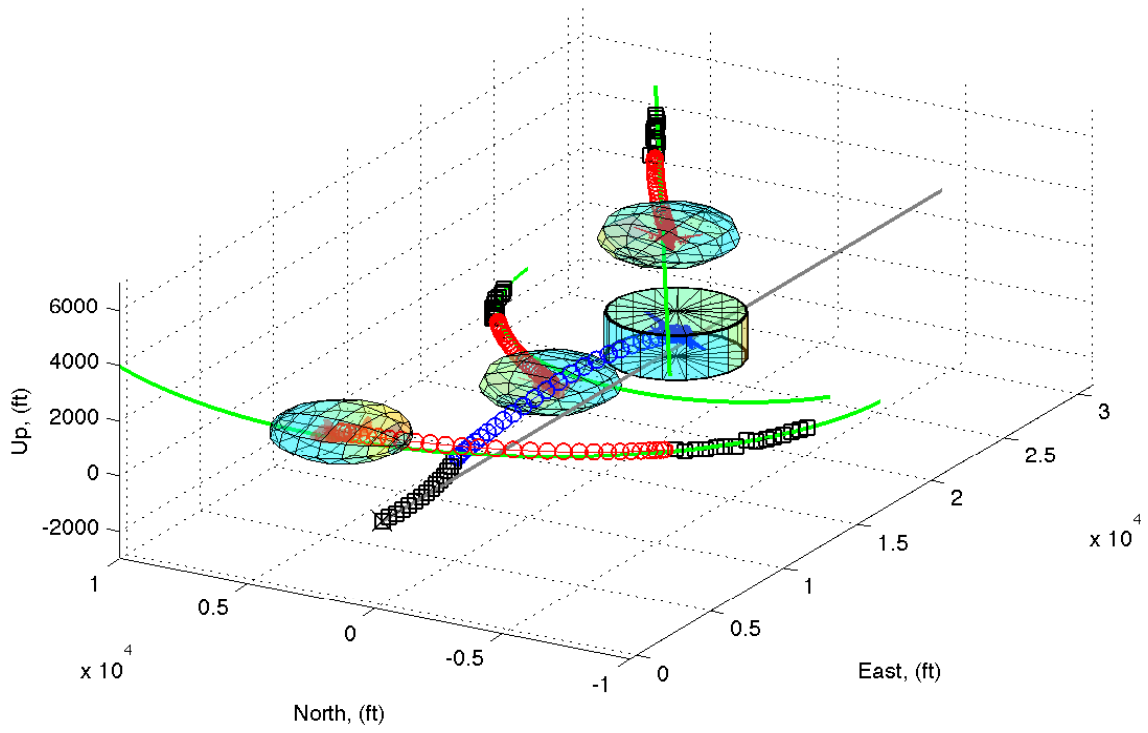
Average Time	111.2 seconds
Shortest Time	31.3 seconds
Longest Time	231.6 seconds

9.6 Cylinder-Ellipsoid Scenario Observations and Conclusions

This demonstration showed the conditional inequality constraint formulation in equation (9.5) is viable; however, this formulation does have potential concerns. Algorithmically, the conditional inequality constraint evaluation appears as:



(a) Top View



(b) Side View

Figure 9.7: Cylinder-Ellipsoid Scenario at Time 45 Seconds

```

if max  $\left( -\ln \left( \left( \frac{\Delta x_{ijk}}{r} \right)^2 + \left( \frac{\Delta y_{ijk}}{r} \right)^2 + \left( \frac{\Delta z_{ijk}}{h} \right)^{100} \right) \right) \leq 0$ 
     $\forall \{ t_i \in [t_0, t_f], k \in [1, (n_e)^2], j \in [1, \dots, \text{number of intruders}] \}$ 
        feasible
    else
        infeasible
end

```

One concern with taking the maximum (or minimum) value in this algorithm is the gradient can be sporadic causing the optimizer problems in finding a search direction. Short of an analytical solution, which does not exist for this problem, one means to minimize this concern is to increase the sampling density in equations (9.7) - (9.9); however, this increase will also increase the NLP execution times.

Another more general concern to the approach herein, which is well-documented in the literature, is that convergence is not guaranteed. This lack of guaranteed convergence is especially problematic when using a receding horizon with stochastic inputs. Due to the stochastic nature, at each time horizon the predicted intruder(s) trajectories will be different than the previous time horizon. Consequently, the coupling of rapidly changing stochastic trajectories for the intruders with highly stringent dynamic constraints for the ownship can result in an infeasible solution even when using the previous converged solution as an initial guess. One method to remedy this concern is by taking advantage of the conditional constraint approach described algorithmically in Section 9.1 of this chapter. This approach analytically calculates in cartesian coordinates the extreme most points on the surface of the intruder's ellipsoidal probability region. The mathematical development for this approach appears in the Appendix C to this document. Using this approach, the NLP solver can be guaranteed a feasible initial guess by specifying an initial path that is greater than the extreme most points on the intruder's ellipsoidal probability surface. A potential concern with this initial guess method is that the guess will influence which local minimum the NLP solver finds; however, this may also be advantageous since the user can now influence the

NLP solver to find the desired local minimum, for instance, a minimum that satisfies certain conditional constraints such as right-of-way.

In summary, this scenario demonstrated that the constraint formulation in equation (9.5) is viable but potentially could have difficulty in finding a search direction due to defining the gradient using the maximum value; increasing the sampling density will mitigate this concern. This demonstration also reinforced the need for the ongoing effort to implement a more appropriate keep-out region around the ownship. An area for future research is to use the conditional constraint formulations presented in the work herein to implement the keep-out regions proposed earlier this year (2014) by the research community to Special Committee-228, the special committee developing operational standards for unmanned aircraft [18, 19]. Another future research area, which spans this entire research effort, is to use the mathematical development presented in this document for finding the extremal points on an intruder's ellipsoidal probability region and use these results to ensure a feasible initial guess for the NLP solver. Since there are multiple extreme paths such as extreme up, extreme down, etc. which will all influence the local minimum the NLP solver finds, an additional research area is to understand how best to pick this initial guess based on satisfying additional conditional constraints such as adhering to right-of-way rules. Ensuring initial guesses are feasible is a rich area for future research. The next section summarizes the research results and identifies potential areas for future research.

X. Conclusions and Recommendations

THIS FINAL chapter summarizes the body of work and presents recommendations for future research based on the findings in this document. The work presented herein successfully expanded on AFRL's SAA efforts by developing (1) techniques for calculating optimal collision avoidance trajectories for RPAs, and (2) techniques for estimating an intruder's trajectory in a stochastic environment. Likewise, the methodology developed and demonstrated in the body of work herein successfully answered the following three questions:

1. **How do you formulate the airborne collision avoidance problem as an optimal control problem?** The work herein demonstrated different cost formulations such as minimum time or minimum path deviation for the airborne collision avoidance problem. The intruders' dynamics were modeled as time-varying probability regions and approximated using minimum volume enclosing ellipsoids which were then interpolated spherically at each collocation node and set as inequality path constraints. In addition, the airborne collision avoidance problem required implementing conditional constraints such as satisfying either a vertical *OR* horizontal separation distance or complying with FAA right-of-way (ROW) rules. These conditional constraints were approximated using either the MAES or sigmoid product methods; both methods achieved reasonable error bounds within the context of the NLP.
2. **How should you model and estimate stochastic intruder(s) for an airborne collision avoidance application in the optimal control problem?** The work herein demonstrated the use of a particle filter to account for nonlinear intruder observation and propagation models within the context of the airborne collision avoidance optimal control problem. The particle filter accurately modeled the intruder's posterior distribution which were then set as inequality path constraints

for the the airborne collision avoidance optimal control problem. In addition, the work presented herein also compared and contrasted the performance of the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and particle filter (PF) for use in an airborne sense and avoid application within the National Airspace System (NAS). The comparison showed the EKF was the most efficient followed closely by the UKF; however, when using a slow measurement update rate such as 1 Hz certain ownship-intruder geometries could result in reduced observability, which would accentuate linearization errors in the EKF and significantly degraded performance. Although the particle filter offered the highest performance in estimating and predicting an intruder’s current and future trajectories for the optimal airborne collision avoidance problem, because of the inherently stochastic and non-deterministic nature, certification by the FAA of a particle filter in a safety-of-flight system for use in the NAS remains an ongoing challenge [52].

3. How do you account for ownship and intruder(s) uncertainties for an airborne collision avoidance application in the optimal control problem?

The work presented demonstrated the use of the novel SLIMVEE algorithm developed herein to capture the time-varying uncertainty regions surrounding the intruder and demonstrated the use of superquadrics to numerically approximate a cylindrical keep-out region around the ownship. Although not implemented in the work presented, the use of superquadrics can be applied to model any unique keep-out region, to include time-varying regions.

In answering these questions, the body of work herein uncovered a number of key findings and recommendations. These key findings and recommendations are presented in following general areas: optimal control and optimal estimation. The following sections list the most significant recommendations (R) in the general order as they appeared in the document starting with optimal control.

10.1 Optimal Control Future Research Recommendations

- A well-known and often-stated limitation of gradient-based NLP search methods is they produce local optimal solutions, which may or may not be global solutions. The formulation and testing of the ROW formulation highlighted the potential applicability of this limitation in the context of airborne collision avoidance. For instance, given identical initial conditions, to enforce a “left turn” constraint required an initial trajectory guess to the left in order for the optimizer to locate the global vice the local optimal solution. A follow-on research effort that explores potential methods for appropriately choosing a “smart” initial guess for complex compounded conditional constraints would be beneficial (R1).
- Another area which would benefit from additional research is to explore the differentiability (steepness) of the gradient for the conditional inequality path constraint in equation (6.19). This remains a particular concern especially when using a small number of fixed collocation nodes, which is likely required in order to satisfy a near real-time implementation. An adaptive node placement strategy could alleviate this concern; however, an adaptive node placement algorithm is not conducive for real-time implementation since NLP convergence times for each time horizon is largely unknown. Thus, an analysis of efficient methods to mitigate against differentiability concerns remains an area for future research (R2).
- Another significant area of concern for a real-time collision avoidance implementation using a direct method is the failure of the NLP solver to find a feasible solution. Lai and Whidborne [90] applied a direct collocation method to solve the optimal control problem for an unmanned obstacle avoidance application. In an optimal airborne collision avoidance application, a failure to converge is a critical area that requires additional study to better understand root causes for potential infeasible solutions, and how best to mitigate against these contingencies in a real-time implementation (R3).

- Comprehensive search algorithms such as genetic algorithms [102] and particle swarm optimization [103] can increase the likelihood of finding the global minimum, yet in the context of the body of work presented, another possible means to achieve a “good” initial guess for the optimal control algorithm is to use the JOCA solution as an initial guess. This approach would utilize the strengths of both algorithms and likely result in a more cost efficient avoidance solution than either of these algorithms by themselves. Thus, developing initial ‘good’ initial guesses is a potential area for future research (R4).
- Another potential area for future research is to study methods to insure the differential constraints are satisfied when interpolating the state dynamics in an receding horizon implementation, which is necessary to establish the appropriate boundary conditions for the next time horizon. Since the node spacing is dense at the beginning of the trajectory where the interpolation occurs, for low dynamic maneuvers this interpolation will likely not violate the differential constraints; however, in more dynamic cases the interpolated states could violate the differential constraints at the boundary condition because the optimizer only enforces equality at the collocation nodes. Exploring methods and conditions to appropriately bound the interpolated states at the boundary conditions is a potential topic for future research (R5).
- Another potential topic for future research is the use of an adaptive time horizon which extends to include predicted trajectories for all intruders that could influence the avoidance solution. Although the limited evaluation in the work herein using an adaptive horizon did not show remarkable savings, there are potential collision avoidance scenarios where an adaptive time horizon could prove especially beneficial. Thus, a research effort to quantify the cost and benefits of an adaptive time horizon could prove beneficial (R6).
- Methods to improve NLP execution times are one of the principal considerations for enabling widespread use of an optimal control approach in real-time collision avoidance

applications. Thus, methods to improve NLP execution times to facilitate near real-time implementation remains an area for future research (R7).

- Without a control penalty the optimal control solution performs an S-turn in the horizontal-plane and a ‘porpoise’ in the vertical-plane about the nominal flightpath trajectory prior to commanding a maximum control maneuver away from the intruder(s). Operationally these maneuvers can lead to excessive fuel consumption and degrade surveillance performance over a target area. Furthermore, these maneuvers are not what air traffic control or other pilots expect from an aircraft operating in the NAS. Therefore, for actual flight implementation a potential future research area is to appropriately scale and then quantify performance differences when applying a control penalty to minimize these oscillations about the nominal trajectory (R8).
- The cylinder-ellipsoid keep-out scenario reinforced the need for the ongoing effort to implement a more appropriate keep-out region around the ownship. An area for future research is to use the conditional constraint formulations presented in the work herein to implement the keep-out regions proposed earlier this year (2014) by the research community to Special Committee-228 (R9).
- Likewise, a future research area is to develop techniques to best incorporate at each time step the extreme most points on an intruder’s ellipsoidal probability region and use these points to ensure a feasible initial guess for the NLP solver (R10).

10.2 Optimal Estimation Future Research Recommendations

- In their seminal work on particle filters, [44] stressed that marginalization “is of outmost importance for high-performance real-time applications.” In the current intruder observation and propagation models, the elevation (vertical) measurement is nonlinear, but the 3-state vertical propagation model is linear, and thus an ideal candidate for marginalization to improve overall filter efficiency. Quantifying performance gains by implementating of a Rao-Blackwell marginalization scheme with

an EKF or UKF for the vertical observation model and a Kalman filter for the vertical dynamics model is an area for future research (R11).

- The work herein briefly explored using a faster than 1 Hz measurement update rate; however, an area for future research is to conduct a more thorough analysis to quantify performance gains and costs resulting from a faster update rate (R12).
- Finally, improving particle filter efficiency to support real-time implementation is an ongoing area for future research (R13).

Although there is still much to be done, this body of work presented a solid method that is practical and represents a good basis for future research. Furthermore, even if real-time implementation is not possible in the near future, the body of work presented can eventually serve as a baseline for evaluating other collision avoidance algorithms. As our national leaders clearly understand, unmanned aircraft will continue to play a vital role in not only our nation's defense but also our nation's economy. The body of work presented herein serves as a solid foundation to build upon to safely integrate remotely piloted aircraft into our national airspace system.

Appendix A:

THIS APPENDIX reviews nearly 20 different problem formulation methods in the literature with potential applicability for an unmanned aircraft collision avoidance application. Even if the methodology presented by the researchers in the literature was not specifically devised for an RPA collision avoidance application (such as piloted *free flight* operations), this review still considered these methods since a logical extension such as an inner-loop controller could allow these methods to work for an RPA collision avoidance application. Likewise, although the focus of this review was on collision avoidance against a moving airborne intruder, this review still considered certain avoidance methods that researchers have implemented specifically for stationary obstacles. The logic for the inclusion of these methods is the same as previous; in certain cases, a logical extension could enable these methods to become applicable for a collision avoidance scenario.

The following review lists the problem formulation methods along with selected references that potentially could support an unmanned aircraft SAA collision avoidance application: Mixed Integer (Non)/Linear Programming [33, 117–121]; Dynamic Programming [62, 122, 123]; Neural Network [30, 37, 124]; Model Predictive Control (MPC)/Receding Horizon Control (RHC) [125–128]; Stochastic Optimal Control [39]; Generalized Polynomial Chaos (gPC) [40, 41]; Collision Cone [129–132]; Geometric [133–136]; Force Field [28]; Markov Decision Process (MDP) [137]; Partially Observable Markov Decision Process (POMDP) [46, 51, 53, 123]; Heuristic [138–140]; Genetic Algorithms [73, 141]; Bayesian Optimal Design [142, 143]; Monte Carlo Simulation [144, 145]; Point of Closest Approach (PCA) [132, 136, 146]; Fuzzy Logic [147]; Proportional Navigation [148]; Game Theory [149, 150]; Optimal Control Problem (OCP) [90, 151–153]; Direct orthogonal collocation [35, 64, 154, 155]; others [156–158].

Table A.1 provides a review of these collision avoidance formulation methods as well as selected references. The far left column lists the overarching problem formulation method and the center column provides a brief description which in general focuses on

the implementation strategy from the most recent reference cited in the far right column. In general the remaining references in the column follow a very similar approach and apply the method in the far left column of the table; however, the table would become too unwieldy and add little additional benefit by attempting to describe the nuance of each reference. Also, some references in the table combine aspects from various techniques. For example, Rathbun et al. [73] primarily used a genetic problem formulation method but they also incorporated a MPC approach to account for uncertainty and generate a real-time trajectory. Therefore, for brevity, the table only list the primary problem formulation method in this and other similar cases.

Table A.1: Survey Results

Survey of Potential RPA Collision Avoidance Formulation Methods		
Formulation Method	Objective: Method Attempts to	References
Collision Cone	This method attempts to use aircraft position, heading, flight path angle and the derivative of these states to determine if the propagated trajectory will result in a violation of a specified minimum miss distance constraint [131]. By definition, a collision cone is the region surrounding an obstacle or intruder where the avoidance algorithm seeks to keep the ownship's velocity vector out of this region. In this formulation an intruder can have more than one collision cone. This approach is valid for both regular and irregularly shaped objects. This formulation performs pairwise avoidance solutions and does not account for uncertainty.	[129–132]
Model Predictive Control (MPC) or Receding Horizon Control (RHC)	This method attempts to minimize uncertainty and facilitate real-time implementation by transitioning an open-loop control scheme into a closed-loop scheme through the use of a finite-time horizon. These methods allow for real-time implementation by calculating a new control at the end of each time horizon. This method does not explicitly account for uncertainty but the iterative nature of the algorithm inherently minimizes the impacts of uncertainty. To improve efficiency, Frew et al. [74] implemented a variable time horizon based on the average uncertainty of the environment. This method provides a concurrent avoidance solution in a multi-intruder environment.	[74, 125–128]

Continued on Next Page...

Survey Results – Continued

Formulation Method	Objective: Method Attempts to	References
Force Field	This method attempts to model an aircraft as a point mass and then treats “each aircraft as a charged particle and use modified electrostatic equations to generate resolution maneuvers” [3]. In this formulation, the method uses the repulsive forces between aircraft to generate collision avoidance maneuvers [3]. Ghosh and Tomlin [159] used a global sink function for each aircraft as a means to attract each aircraft towards its desired destination and they applied repulsive and vortex potential fields about each aircraft to ensure collision avoidance. A downside to this method is “the attractive and repulsive forces might cancel each other and lead to a zero resultant force” necessitating the need for “higher-level planners to escape from such trap” [54]. These methods do not perform concurrent avoidance solutions.	[28, 159]
Markov Decision Process (MDP)/Partially Observable Markov Decision Process (POMDP)	This method attempts to model a “stochastic process where the state of the system changes probabilistically according to the current state and action. MDPs assume that the state is fully observable. POMDPs remove that assumption and replace it with a stochastic model for observations, and hence they have more expressive power” [137]. A MDP/POMDP are typically solved via dynamic programming. Often stated critique of these methods is the curse of dimensionality that makes exact solutions “computationally intractable in general” [46]. In an effort to update the TCAS logic, Kochenderfer et al. proposed an innovative POMDP approach that takes advantage of pre-computing and storing results off-line to in order to facilitate real-time operations [123]. These methods often employ pairwise solutions.	[46, 51, 53, 123, 137]
Mixed-Integer (Non)/Linear Programming	This method attempts to find an optimal solution by minimizing a cost function such as aircraft separation distance subject to various nonlinear (or linear) constraints. This methods does not inherently account for uncertainty in the problem formulation. This method provides pairwise collision avoidance solutions in a multi-intruder environment.	[33, 117– 121]

Continued on Next Page...

Survey Results – Continued

Formulation Method	Objective: Method Attempts to	References
Dynamic Programming	This method attempts to apply a sequential decision process [160] based on Bellman’s Principle of Optimality to solve subparts of an overall problem and then combines these results to obtain an optimal solution to the complete problem. This method allows system designers to compute and then store results offline to facilitate real-time implementation. This method provides pairwise avoidance solutions. Although this method does not inherently account for uncertainty, POMDP and MDP which are dynamic programming methods explicitly account for uncertainty.	[62, 122, 123]
Neural Network	This method attempts to learn and adapt to the environment. The implementation in collision avoidance applications are varied. For example, Horn et al.[37] applied this method to reduce “computational requirements by removing the need for collocation and providing fast computation of gradients when compared with direct and pseudospectral collocation methods.” In that implementation this method reduced computational cost and generated RPA trajectories comparable to those produced by direct collocation and pseudospectral methods. This method can be used to generate concurrent avoidance solution; however, Wang et al. [30] did not formulate the avoidance solution in that manner.	[30, 37, 124]
Stochastic Optimal Control	This method attempts to explicitly account for and minimize the expected value of state uncertainty in the formulation of the OCP. Liu and Hwang [39] modeled the aircraft dynamics as a Stochastic Differential Equation (SDE) and applied this method to generate an optimal conflict free feedback control law; they solved the stochastic OCP numerically using a Markov chain approximation. This method requires more computational complexity than the deterministic OCP formulation but preserves the same functionality.	[39]
Generalized Polynomial Chaos (gPC)	This method attempts to account for state uncertainty by transforming the stochastic trajectory OCP into a deterministic OCP and then solving this resulting deterministic OCP via direct orthogonal collocation methods [41]. The gPC method is computationally intensive. Due to the large computationally requirements, the references cited did not use this formulation as a real-time obstacle avoidance planner but instead used this method during mission planning to determine an <i>a priori</i> path through an area with assumed fixed obstacles with position uncertainty.	[40, 41]

Continued on Next Page...

Survey Results – Continued

Formulation Method	Objective: Method Attempts to	References
Geometric	This method attempts to utilize simple geometric relations and treats all aircraft as point masses to obtain closed-form analytical solutions that keep the ownship outside of a set boundary from an intruder aircraft. Bilimoria [133] defined an optimal collision avoidance solution as one that minimized the ownship’s velocity vector changes resulting in a minimum path deviations from the nominal trajectory. This formulation by design performs sequential pairwise solutions which leads to sub-optimal avoidance solutions in a multi-intruder environment since the multi-intruder solution does not minimize deviations from nominal path [133]. Further, this formulation does not account for uncertainty.	[133–136]
Genetic Algorithms	This method attempts to employ “a search algorithm based on the conjecture of natural selection and genetics” [161] . In general, the algorithm’s multi-path search tends to reduce the possibility of local “minimum trapping” and there is no need for computation of derivatives or other auxiliary functions [161]. This method “explores the search space where the probability of finding improved performance is high”[161]. Durand et al. notes that although Genetic algorithms are very efficient in solving “global combinatorial optimization problem”, they are not very efficient in solving local search areas with “good precision” [141].	[73, 141]
Bayesian Optimal Design	This method attempts to solve a non-convex optimization problem while modeling stochastic processes such as wind disturbances on an aircraft. When applied using a Markov system, as done in the cited references, an alternate name for this formulation method is a particle filter, which this document describes in detail in Chapters II and III. Kantas et al. [142] attempted to minimize time to a waypoint while avoiding other aircraft by a set separation distance. Tirri et al. [50] used a particle filter to improve intruder state estimate as compared to an Extended Kalman Filter (EKF) and they also used the particle filter to assess the collision risk potential by estimating the distance at the closest point of approach to the intruder. Disadvantages of particle filters are they require intensive computation overhead, often making real-time implementation challenging.	[50, 142, 143]

Continued on Next Page...

Survey Results – Continued

Formulation Method	Objective: Method Attempts to	References
Monte Carlo Simulation	<p>This method attempts to use probabilistic models to formulate conflict resolution as the optimization of an expected value criterion with probabilistic constraints [144]. Lecchini et al. [144], applied a Monte Carlo framework to account for uncertainty in an air traffic control collision avoidance application where the penalty function guaranteed constraint satisfaction but delivered suboptimal solutions. This implementation used simulation to calculate the probability of conflict for various pairwise combinations of aircraft descending into a traffic area to determine appropriate time separation to reduce the potential of a conflict. Prandini et al. [145] developed closed-form approximations for the probability of conflict for a pairwise aircraft encounter and found favorable results when using Monte Carlo simulations to demonstrate the effectiveness of this approximation to generate safe resolution maneuvers; however, the authors discovered potential problems with this approach in a congested multiple aircraft environment.</p>	[144, 145]
Point of Closest Approach (PCA)	<p>This method attempts to identify the worst-case conflict condition between a pair of aircraft. Krozel and Peters [162] showed that for the 2D case, at the PCA between two aircraft the miss vector and the relative motion vector are orthogonal allowing for the calculation of the time-to-closest approach which researchers have used to design collision avoidance algorithms. For instance, assuming a completely cooperative environment, Park et al. [136] applied this method to develop a cooperative avoidance maneuver known as “Vector Sharing Resolution” where two conflicting RPAs cooperatively maneuver to “share the conflict region.” The limitation of this method is that it performs pairwise solutions; to account for this limitation Park et al. proposed identifying an artificial center between two intruders in a multi-intruder environment.</p>	[132, 136, 146]

Continued on Next Page...

Survey Results – Continued

Formulation Method	Objective: Method Attempts to	References
Graph search methods	These methods attempt to use search algorithms to find feasible paths that are conflict free. Mujumdar and Padhi [25] note there are numerous graph search algorithms such the deterministic A^* search algorithm, the Voronoi graph search algorithm, and algorithms such as the Probabilistic Roadmap Method that consider uncertainty. Although mission planners mainly use these algorithms to calculate an <i>a priori</i> path, researchers have modified these algorithms to provide reactive or local collision avoidance solutions [25]. For instance Hwangbo et al. [26] applied a coarse global search algorithm to determine a “kinematically feasible obstacle-free path in a discretized 3D workspace” and then used a “fine” local search algorithm to “compute a more accurate trajectory” for the RPA to follow. The authors continued to iterate on this two-phase search scheme and demonstrated real-time performance in simulation. One of the downside of this implementation method is the significant memory storage requirement [25].	[26]
Fuzzy Logic	This method attempts to apply an “if-then” construct based on heuristic information to develop a collision avoidance solution [147]. Dong et al. [147] developed and applied fuzzy logic control algorithms in simulation to demonstrate the ability for a RPA to track a pre-planned path while avoiding unexpected obstacles. In their simple 2D simulation, the authors demonstrated the ability for the RPA to avoid multiple obstacles of varying shapes; however, this method did not minimize a cost function nor achieve an optimal avoidance solution.	[147, 163]
Proportional Navigation	This method attempts to minimize the line-of-sight rate between the ownship and an intruder [132]. This method has its genesis in popular strategies for missile engagement scenarios [148]. Han and Bang [148] applied this method in simulation for an unmanned aircraft application. They first discussed sufficient conditions for collision avoidance based on pairwise geometric relationships and then attempted to design an optimal collision avoidance algorithm. Their simulations results were only in 2D and the results did not consider a multi-intruder environment. Further, the results did not discuss the optimality of the avoidance solutions.	[148]

Continued on Next Page...

Survey Results – Continued

Formulation Method	Objective: Method Attempts to	References
Game Theory	This method attempts to provide collision avoidance solution based on a cooperative environment between the ownship and intruder(s). Researchers have used various implementation schemes for this method. For instance, Cruck and Lygeros [150] formulated the conflict situation between the ownship and the intruder as a generalized pursuit-evasion game where the ownship's goal was to ensure separation. The references cited only demonstrated results using computer-based simulations; however, in general, this method provides suboptimal solutions.	[149, 150, 164]
Optimal Control Problem (OCP)	This method attempts to find the optimal control that minimizes (or maximizes) a cost function while satisfying differential and algebraic constraints for the system. In general, this method requires the researcher to derive the first-order necessary conditions for optimality via the calculus of variations and Pontryagin's minimum principle [63]. This formulation method inherently does not account for state uncertainty in the problem formulation.	[90, 151–153]
OCP via Direct Collocation	This method attempts to solve the continuous time optimal control problem by using orthogonal basis functions that satisfy the states, controls, differential, and algebraic constraints exactly at discrete collocation nodes. Essentially, this method transcribes an OCP to a Nonlinear Programming (NLP) problem via parametrization and discretization and then solves the NLP using well developed algorithms [90]. This method allows for local collision avoidance in 3D for a fixed-wing RPA both in a non-cooperative and cooperative environment while optimizing towards a global goal; this method performs optimal concurrent collision avoidance and provides a discrete cost function to assess optimality.	[35, 64, 154, 155]
Other	These methods attempt to achieve optimal or suboptimal avoidance solutions via ad hoc means or methods that do not fit conveniently into other previously defined formulation methods (e.g. heuristic satisficing strategies). Other formulation methods not explicitly included in this table are leader-follower methods which are popular in RPA formation flight implementations and heuristic algorithms such as evolutionary based algorithms that attempt to evolve by learning from previous system outcomes.	[138–140, 156–158]

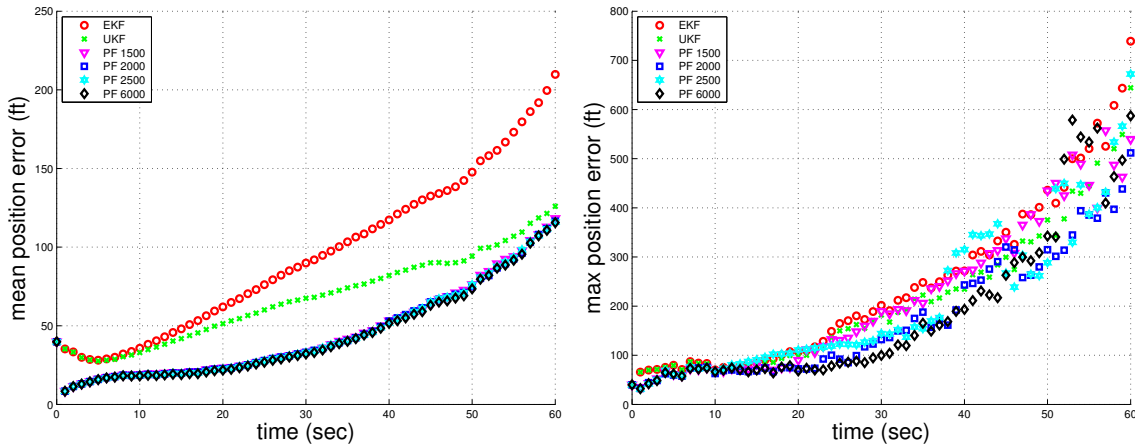
Appendix B:

THIS APPENDIX presents the remaining results for the nonlinear filter evaluation from Section 8.7. These results were not included in the main body of this document since these results merely repeated the trend as seen in the earlier results. However, for completeness, these results are included here.

B.1 Scenario Two

B.1.1 Scenario Two, Case Two: Half Standard-Rate Turn.

This 2D case was not a stressing scenario. The initial set up for this scenario was identical to Scenario One, Case Two with the sole exception the intruder's initial heading was now at a 90° angle to the ownship. At the start of this scenario the intruder was approximately 5.25 NM east (x -axis) and approximately 1.89 NM north (y -axis) of the ownship. The intruder maintained a half standard-rate counterclockwise turn to the south and the scenario terminated with the intruder abeam the ownship separated by a slant range of 2460 feet. The results of this scenario appear in Figure B.1 where the mean of the



(a) Mean Error for Half Standard-Rate Turn

(b) Worst-Case Error for Half Standard-Rate Turn

Figure B.1: Mean & Worst-Case Position Error for Half Standard-Rate Turn

position errors appear in panel (a) and the maximum position errors in panel (b). For this scenario, none of the filter's position errors were over the 820-foot threshold.

Figure B.2 shows the mean and maximum turn-rate errors for all three filters. Again, without a priori knowledge of the intruder's maneuvers, the initial mean estimate for the intruder's turn-rate state was zero. Since the intruder in this scenario performed a half standard-rate turn, the initial turn-rate error for all three filters was approximately 1.5 degrees.

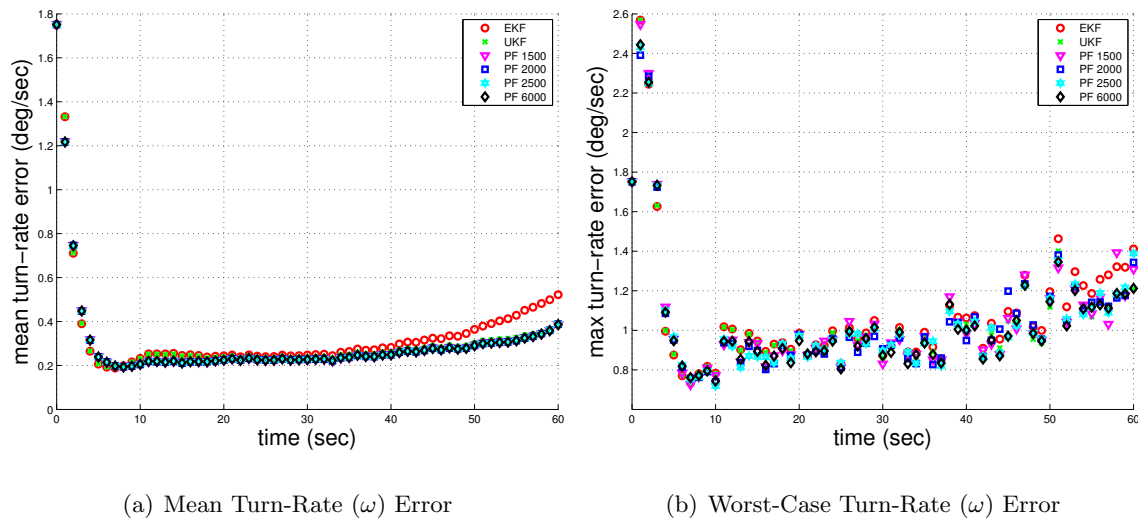


Figure B.2: Mean & Worst-Case Turn-Rate Error for Half Standard-Rate Turn

B.2 Scenario Three

B.3 Scenario Three, Case Two: Half Standard-Rate Turn

The initial set up for this 3D case was identical to Scenario One, Case Two with the sole exception the intruder now descended at a rate of 1250 ft/min. At the start of this scenario the intruder was approximately 5.25 NM east (x -axis), approximately 1.89 NM north (y -axis), and co-altitude with the ownship. In addition to the 1250 ft/min descent rate, the intruder also maintained a half standard-rate counterclockwise turn to the south. The scenario terminated with the intruder directly abeam the ownship separated

horizontally by 2460 feet and vertically by 1250 feet. The results of this scenario appear in Figure B.3. For this scenario, none of the filter's position errors were over the 820-foot threshold. Figure B.4 shows the turn-rate errors for all three filters.

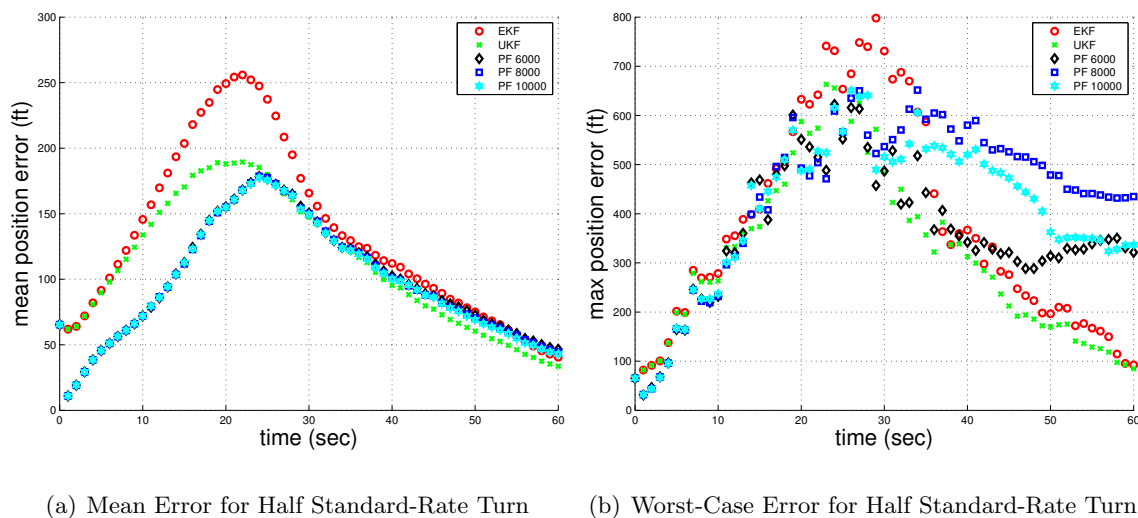


Figure B.3: Mean & Worst-Case Position Error for Half Standard-Rate Turn

Like the previous scenarios, the initial mean estimate for the intruder's turn-rate was zero.

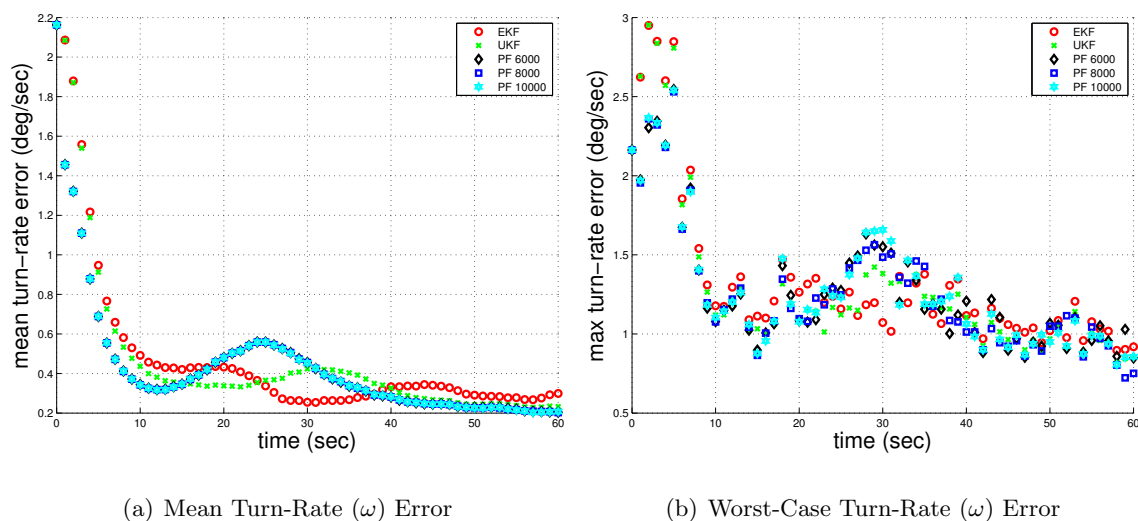


Figure B.4: Mean & Worst-Case Turn-Rate Error for Half Standard-Rate Turn

B.4 Scenario Three, Case Three: No Turn

The set up for this final 3D case was identical to Scenario One, Case Three with the exception the intruder now descended at a rate of 1,250 ft/min. At the start of the scenario both aircraft flew towards each with the intruder starting approximately 6.28 NM east (x -axis) and 0.2 NM north (y -axis) of the ownship. The intruder did not turn but maintained a west heading and the scenario terminated with the two aircraft facing each other separated horizontally by 2460 feet and vertically by 1250 feet. The results of this scenario appear in Figure B.5. For this scenario none of the filter's position errors were over the 820-foot threshold.

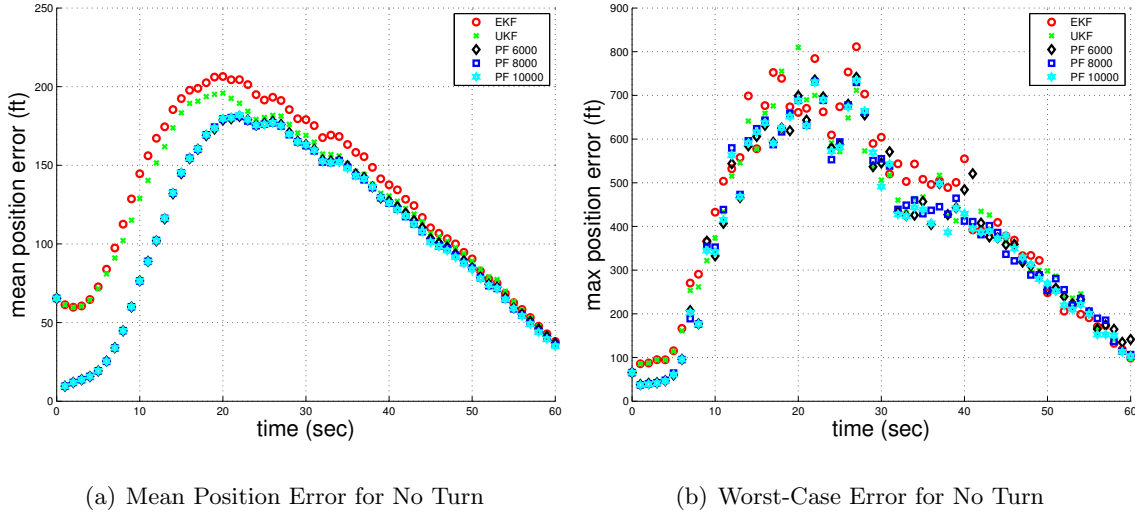
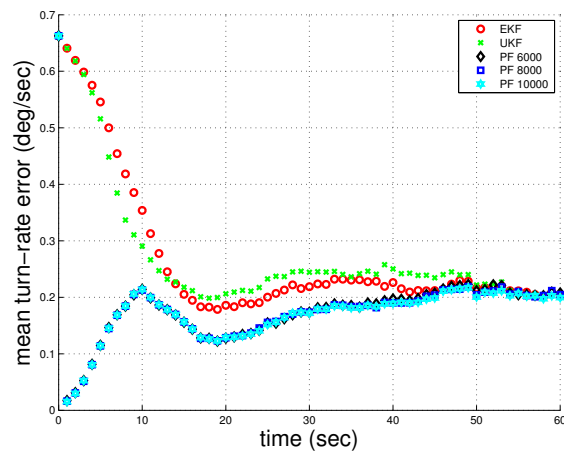
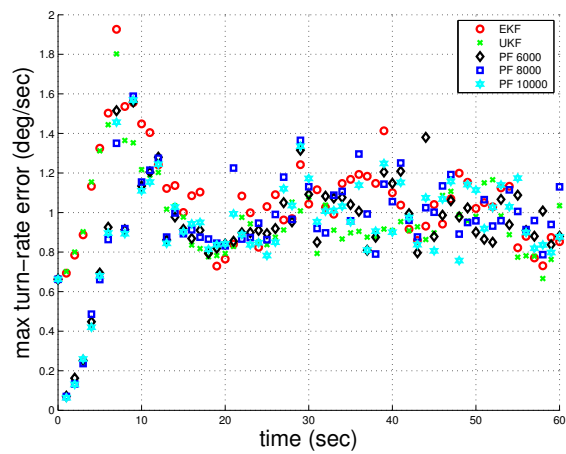


Figure B.5: Mean & Worst-Case Position Error for No Turn

Figure B.6 shows the mean and maximum turn-rate errors for all three filters. The initial mean estimate for the intruder's turn-rate state was zero. Since the intruder in this scenario did not turn, the errors were small for all three filters.



(a) Mean Turn-Rate (ω) Error



(b) Worst-Case Turn-Rate (ω) Error

Figure B.6: Mean & Worst-Case Turn-Rate Error for No Turn

Appendix C:

THIS APPENDIX presents the mathematical development described in Section 9.1 to analytically determine the uppermost and lowermost points on the surface of an ellipsoidal probability region to use as an efficient check for feasibility. Repeating the equation here for convenience, the equation of an ellipsoid (E) in quadratic form appears as:

$$E(\mathbf{x}) = (\mathbf{x} - \mathbf{c}_e)^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) \leq 1 \quad (\text{C.1})$$

where \mathbf{A} is a positive definite symmetric matrix and \mathbf{c}_e is the center of the ellipsoid. The equation for a cylinder (C) with height h and radius r appear as,

$$x^2 + y^2 \leq r^2 \quad \text{where } |z| \leq h \quad (\text{C.2})$$

thus,

$$E = \{\mathbf{x} \in \mathbb{R}^3 : E(\mathbf{x}) = (\mathbf{x} - \mathbf{c}_e)^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) \leq 1\} \quad (\text{C.3})$$

$$C = \{\mathbf{x} \in \mathbb{R}^3 : x^2 + y^2 \leq r^2 \quad \text{where } |z| \leq h\} \quad (\text{C.4})$$

$$(\text{C.5})$$

The top and bottom of the cylinder (C) define bounding planes in the z -axis for the inequality path constraint such that the surface of the ellipsoid can either pass entirely above or below the height of the cylinder for a feasible trajectory. Therefore, the first step is to locate the highest and lowest points in the z -axis of the ellipsoid, that is, find $\arg \min (E(\mathbf{x}) = 1)$. This is done by using the implicit function theorem where,

$$F(\mathbf{x}) = E(\mathbf{x}) - 1 = 0 \quad (\text{C.6})$$

$$\implies z = f(x, y) \quad (\text{C.7})$$

Therefore,

$$0 = \frac{\partial E}{\partial x} = 2\mathbf{e}_1^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) + 2\mathbf{e}_3^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) \frac{\partial f}{\partial x} \quad (\text{C.8})$$

$$0 = \frac{\partial E}{\partial y} = 2\mathbf{e}_2^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) + 2\mathbf{e}_3^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) \frac{\partial f}{\partial y} \quad (\text{C.9})$$

which implies,

$$\mathbf{e}_3^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) \frac{\partial f}{\partial x} = -\mathbf{e}_1^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) \quad (\text{C.10})$$

$$\mathbf{e}_3^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) \frac{\partial f}{\partial y} = -\mathbf{e}_2^T \mathbf{A} (\mathbf{x} - \mathbf{c}_e) \quad (\text{C.11})$$

where \mathbf{e}_i denote unit vectors in the x, y, z directions, respectively. In order for

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y} = 0 \quad (\text{C.12})$$

requires that $\mathbf{e}_i \mathbf{A} (\mathbf{x} - \mathbf{c}_e) = 0$. Thus,

$$\sum_{j=1}^3 \mathbf{A}_{1j} (\mathbf{x} - \mathbf{c}_{e_j}) = 0 \quad (\text{C.13})$$

$$\sum_{j=1}^3 \mathbf{A}_{2j} (\mathbf{x} - \mathbf{c}_{e_j}) = 0 \quad (\text{C.14})$$

Therefore, in order to satisfy equation (C.1),

$$1 = (z - c_{e3}) \sum_{j=1}^3 \mathbf{A}_{3j} (\mathbf{x} - \mathbf{c}_{e_j}) = 0 \quad (\text{C.15})$$

Equations (C.13) - (C.15) determine the minimum and maximum z of the function $z = f(x, y)$. Since equation (C.13) are planes in \mathbb{R}^3 with the point \mathbf{c}_e on both planes. The intersection of these planes is a line defined by

$$\zeta(t) = \mathbf{c}_e + t\beta \quad (\text{C.16})$$

where β is the direction vector given by,

$$\beta = \begin{bmatrix} \hat{i} & \hat{j} & \hat{k} \\ \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{A}_{13} \\ \mathbf{A}_{12} & \mathbf{A}_{22} & \mathbf{A}_{23} \end{bmatrix} \quad (\text{C.17})$$

$$\begin{aligned} & (\mathbf{A}_{12}\mathbf{A}_{23} - \mathbf{A}_{22}\mathbf{A}_{13}) \hat{i} + \\ & = (\mathbf{A}_{12}\mathbf{A}_{13} - \mathbf{A}_{11}\mathbf{A}_{23}) \hat{j} + \\ & \quad (\mathbf{A}_{11}\mathbf{A}_{22} - \mathbf{A}_{12}^2) \hat{k} \end{aligned} \quad (\text{C.18})$$

To satisfy equation (C.15), need to determine the value t such that

$$(\zeta(t) - c_{e_3}) \sum_{j=1}^3 \mathbf{A}_{3j} (\zeta(t) - c_{e_3}) = 1 \quad (\text{C.19})$$

or

$$t\beta_3 \sum_{j=1}^3 \mathbf{A}_{3j} \beta_j t = 1 \quad (\text{C.20})$$

where $\beta_3 = (\mathbf{A}_{11}\mathbf{A}_{22} - \mathbf{A}_{12}^2)$. Thus,

$$t^2 = \left[\beta_3 \sum_{j=1}^3 \mathbf{A}_{3j} \beta_j \right]^{-1} \quad (\text{C.21})$$

$$t = \pm \sqrt{\frac{\beta_3}{\det \mathbf{A}}} \quad (\text{C.22})$$

Therefore, the minimum and maximum altitude, or z -axis, of ellipsoid appears as:

$$z_{\min} = c_{e_3} - \sqrt{\frac{\beta_3}{\det \mathbf{A}}} \quad (\text{C.23})$$

$$z_{\max} = c_{e_3} + \sqrt{\frac{\beta_3}{\det \mathbf{A}}} \quad (\text{C.24})$$

Repeating the same methodology in equations (C.6) - (C.19), the minimum and maximum x and y axis values of ellipsoid appears are:

$$x_{\min, \max} = c_{e_1} \pm \sqrt{\frac{\beta_1}{\det \mathbf{A}}} \quad (\text{C.25})$$

$$y_{\min, \max} = c_{e_2} \pm \sqrt{\frac{\beta_2}{\det \mathbf{A}}} \quad (\text{C.26})$$

Bibliography

- [1] Unmanned Aircraft System Task Force Airspace Integration Integrated Production Team, “Department of Defense Unmanned Aircraft System Airspace Integration Plan, Version 2.0.” Distribution Statement A: Approved for Public Release, March 2011.
- [2] Northrop Grumman Aerospace Systems, “Design Description Document (DDD) for Multi-Sensor Integrated Conflict Avoidance (MuSICA),” Contract Name: Air Vehicle Technology Insertion Program (AVTIP), Contracting Agency: Air Force Research Laboratory, AFRL/RQQC, Wright-Patterson Air Force Base, May 2010.
- [3] J. K. Kuchar and L. C. Yang, “A Review of Conflict Detection and Resolution Modeling Methods,” *Intelligent Transportation Systems, IEEE Transactions on*, vol. 1, no. 4, pp. 179–189, 2000.
- [4] A. Mujumdar and R. Padhi, “Evolving Philosophies on Autonomous Obstacle/Collision Avoidance of Unmanned Aerial Vehicles,” *Journal of Aerospace Computing, Information, and Communication*, vol. 8, no. 2, pp. 17–41, 2011.
- [5] E. W. Weisstein, “Point-Line Distance–3-Dimensional From MathWorld–A Wolfram Web Resource. <http://mathworld.wolfram.com/Point-LineDistance3-Dimensional.html>,” November 2013.
- [6] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House Publishers, 2004.
- [7] A. Doucet, N. De Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. Springer, 2001.
- [8] A. Jaklic, A. Leonardis, and F. Solina, *Segmentation and Recovery of Superquadrics*, vol. 20. Springer, 2000.
- [9] R. H. Chen, A. Gevorkian, A. Fung, and W.-Z. Chen, “Multi-Sensor Data Integration for Autonomous Sense and Avoid,” *AIAA Infotech at Aerospace*, 2011.
- [10] J. Horgan, “The Drones Come Home,” *National Geographic*, vol. 223, pp. 122–135, March 2013.
- [11] A. M. Dolan and R. M. Thompson II, “Integration of Drones into Domestic Airspace: Selected Legal Issues,” Prepared for Members and Committees of Congress, Congressional Research Service: Report for Congress, April 2013.
- [12] T. V. Brook, “Air Force to Train More on Drones,” June 2009.
- [13] Conference Report to Accompany H.R. 658, *FAA Modernization And Reform Act of 2012*. 112th Congress 2d Session, House of Representatives, Report 112-381, U.S. Government Printing Office, Washington DC, February 2012.

- [14] FAA Sponsored Sense and Avoid Workshop, “Sense and Avoid (SAA) for Unmanned Aircraft Systems (UAS) Second Caucus Workshop Report January 18, 2013,” tech. rep., Federal Aviation Administration, 2013.
- [15] Office of Secretary of Defense, “The Unmanned Systems Integrated Roadmap FY2011-2036,” Approved for Open Publication. Reference Number: 11-S-3613, 2011.
- [16] Federal Aviation Administration, “Certificate of Authorization or Waiver (COA).”
- [17] G. Warwick, “Sense-And-Avoid System to Transition to Global Hawk,” *Aviation Week*, June 2012.
- [18] The Radio Technical Commission for Aeronautics (RTCA), “<http://www.rtca.org>.”
- [19] B. Carey, “New RTCA Committee Seeks to Expedite UAS Standards.” at <http://www.ainonline.com/aviation-news/2013-04-05/new-rtca-committee-seeks-expedite-uas-standards>, April 2013.
- [20] D. Maroney, R. Bolling, R. Athale, and A. Christiansen, “Experimentally Scoping the Range of UAS Sense and Avoid Capability,” in *2nd AIAA Infotech@ Aerospace Conference and Exhibit*, 2007.
- [21] Federal Aviation Administration, “Introduction to TCAS II Version 7.1.” booklet, United States Department of Transportation, February 2011.
- [22] Federal Aviation Administration, “Automatic Dependent Surveillance-Broadcast (ADS-B).” at <http://www.faa.gov/nextgen/implementation/programs/adsb>, November 2011.
- [23] J. Utt, J. McCalmont, and M. Deschenes, “Development of a sense and avoid system,” *AIAA Infotech at Aerospace*, 2005.
- [24] L. Scally and M. Bonato, *Unmanned Sense and Avoid Radar (USTAR)*. American Institute of Aeronautics and Astronautics, March 2011.
- [25] A. Mujumdar and R. Padhi, “Nonlinear Geometric and Differential Geometric Guidance of UAVs for Reactive Collision Avoidance,” tech. rep., DTIC Document, 2009.
- [26] M. Hwangbo, J. Kuffner, and T. Kanade, “Efficient two-phase 3d motion planning for small fixed-wing uavs,” in *Robotics and Automation, 2007 IEEE International Conference on*, pp. 1035–1041, 2007.
- [27] J. N. Amin, J. D. Boskovic, and R. K. Mehra, “A fast and efficient approach to path planning for unmanned vehicles,” in *Proceedings of AIAA Guidance, Navigation, and Control Conference and Exhibit*, pp. 21–24, 2006.
- [28] T. Paul, T. R. Krogstad, and J. T. Gravdahl, “Modelling of UAV Formation Flight Using 3D Potential Field,” *Simulation Modelling Practice and Theory*, vol. 16, no. 9, pp. 1453–1462, 2008.

- [29] Y. Watanabe, A. J. Calise, and E. N. Johnson, "Vision-based obstacle avoidance for uavs," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007.
- [30] X. Wang, V. Yadav, and S. Balakrishnan, "Cooperative UAV Formation Flying with obstacle/collision avoidance," *Control Systems Technology, IEEE Transactions on*, vol. 15, no. 4, pp. 672–679, 2007.
- [31] T. Williamson and N. Spencer, "Development and Operation of the Traffic Alert and Collision Avoidance System (TCAS)," *Proceedings of the IEEE*, vol. 77, no. 11, pp. 1735–1744, 1989.
- [32] J. Krozel, T. Mueller, and G. Hunter, *Free Flight Conflict Detection and Resolution Analysis*. American Institute of Aeronautics and Astronautics, 2013/05/13 1996.
- [33] M. Christodoulou and S. Kodaxakis, "Automatic Commercial Aircraft-Collision Avoidance in Free Flight: The Three-Dimensional Problem," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 7, no. 2, pp. 242–249, 2006.
- [34] C. Goerzen, Z. Kong, and B. Mettler, "A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance," *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1-4, pp. 65–100, 2010.
- [35] A. U. Raghunathan, V. Gopal, D. Subramanian, L. T. Biegler, and T. Samad, "Dynamic Optimization Strategies for Three-Dimensional Conflict Resolution of Multiple Aircraft," *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 4, pp. 586–594, 2004.
- [36] A. J. Eele and A. Richards, "Path-Planning with Avoidance Using Nonlinear Branch-and-Bound Optimization," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 2, pp. 384–394, 2009.
- [37] J. F. Horn, E. M. Schmidt, B. R. Geiger, and M. P. DeAngelo, "Neural Network-Based Trajectory Optimization for Unmanned Aerial Vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 2, pp. 548–562, 2012.
- [38] B. R. Geiger, J. F. Horn, G. L. Sinsley, J. A. Ross, L. N. Long, and A. F. Niessner, "Flight Testing a Real-Time Direct Collocation Path Planner," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 6, pp. 1575–1586, 2008.
- [39] W. Liu and I. Hwang, "Probabilistic Aircraft Conflict Resolution: A Stochastic Optimal Control Approach," in *2012 American Control Conference (ACC) Fairmont Queen Elizabeth, Montreal, Canada. 27 - 29 June*, pp. 1714–1719, 2012.
- [40] G. C. Cottrill, *Hybrid Solution of Stochastic Optimal Control Problems Using Gauss Pseudospectral Method and Generalized Polynomial Chaos Algorithms*. PhD thesis, Air Force Institute of Technology, 2012.
- [41] H. F. Liu, Y. Zhang, S. F. Chen, and J. Chen, "Autonomous Vehicle Trajectory Planning under Uncertainty Using Stochastic Collocation," *Advanced Materials Research*, vol. 580, pp. 175–179, 2012.

- [42] F. Daum, “Nonlinear Filters: Beyond the Kalman Filter,” *Aerospace and Electronic Systems Magazine, IEEE*, vol. 20, no. 8, pp. 57–69, 2005.
- [43] J. Jansson and F. Gustafsson, “A Framework and Automotive Application of Collision Avoidance Decision Making,” *Automatica*, vol. 44, no. 9, pp. 2347 – 2351, 2008.
- [44] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, “Particle Filters for Positioning, Navigation, and Tracking,” *Signal Processing, IEEE Transactions on*, vol. 50, no. 2, pp. 425–437, 2002.
- [45] T. Jones, “Tractable Conflict Risk Accumulation in Quadratic Space for Autonomous Vehicles,” *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 1, pp. 39–48, 2006.
- [46] J. P. Chryssanthacopoulos and M. J. Kochenderfer, “Accounting for State Uncertainty in Collision Avoidance,” *Journal of Guidance, Control, and Dynamics*, vol. 34, pp. 951–960, July - August 2011.
- [47] L. C. Yang, *Aircraft Conflict Analysis and Real-Time Conflict Probing Using Probabilistic Trajectory Modeling*. PhD thesis, Massachusetts Institute of Technology, 2000.
- [48] P. K. M. Jinwhan Kim, S. S. Vaddi, “Comparison between Three Spiraling Ballistic Missile State Estimators,” vol. Copyright © 2008 by Optimal Synthesis Inc. Published by the AIAA, Inc., with permission., 8 - 21 August 2008.
- [49] D. Alejo, R. Conde, J. Cobano, and A. Ollero, “Multi-UAV Collision Avoidance With Separation Assurance Under Uncertainties,” in *Mechatronics, 2009. ICM 2009. IEEE International Conference on*, pp. 1–6, 2009.
- [50] A. E. Tirri, G. Fasano, D. Accardo, and A. Moccia, “Airborne Tracking Based on Particle Filtering for UAS Sense and Avoid,” in *Infotech@Aerospace 2012 Conference*, AIAA, 19 - 21 June 2012, Garden Grove, California.
- [51] T. Wolf and M. Kochenderfer, “Aircraft Collision Avoidance Using Monte Carlo Real-Time Belief Space Search,” *Journal of Intelligent & Robotic Systems*, vol. 64, no. 2, pp. 277–298, 2011.
- [52] D. M. Asmar, M. J. Kochenderfer, and J. P. Chryssanthacopoulos, “Vertical State Estimation for Aircraft Collision Avoidance with Quantized Measurements,” *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 6, pp. 1797–1802, 2013.
- [53] H. Bai, D. Hsu, M. J. Kochenderfer, and W. S. Lee, “Unmanned aircraft collision avoidance using continuous-state pomdps,” *Robotics: Science and Systems VII*, p. 1, 2012.
- [54] S. Temizer, *Planning Under Uncertainty for Dynamic Collision Avoidance*. PhD thesis, Massachusetts Institute of Technology, 2011.
- [55] D. E. Kirk, *Optimal Control Theory: An Introduction*. Courier Dover Publications, 2012.

- [56] J. P. Chryssanthacopoulos and M. J. Kochenderfer, "Decomposition Methods for Optimized Collision Avoidance with Multiple Threats," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 2, pp. 398–405, 2012.
- [57] J. J. Rebollo, A. Ollero, and I. Maza, "Collision Avoidance Among Multiple Aerial Robots and Other Non-Cooperative Aircraft Based on Velocity Planning," in *Proceedings of ROBOTICA 2007 Conference, Paderne, Portugal*, 2007.
- [58] S. S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Artech House, 1999.
- [59] X. Li and V. Jilkov, "Survey of Maneuvering Target Tracking. Part I. Dynamic Models," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 39, no. 4, pp. 1333–1364, 2003.
- [60] X. Li and V. Jilkov, "Survey of Maneuvering Target Tracking. Part V. Multiple-Model Methods," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 41, no. 4, pp. 1255–1321, 2005.
- [61] R. Singer, "Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. AES-6, no. 4, pp. 473–483, 1970.
- [62] Y. Ikeda, B. Nguyen, A. Barfield, B. Sundqvist, and S. Jones, "Automatic air collision avoidance system," in *SICE 2002. Proceedings of the 41st SICE Annual Conference*, vol. 1, pp. 630–635 vol.1, 2002.
- [63] G. T. Huntington, *Advancement and Analysis of a Gauss Pseudospectral Transcription for Optimal Control Problems*. PhD thesis, Citeseer, 2007.
- [64] Q. Gong, R. Lewis, and M. Ross, "Pseudospectral Motion Planning for Autonomous Vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 3, pp. 1039–1045, 2009.
- [65] K. Horie and B. A. Conway, "Optimization for fighter aircraft vertical-plane maneuvering using poststall flight," *Journal of aircraft*, vol. 37, no. 6, pp. 1017–1021, 2000.
- [66] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of guidance, control, and dynamics*, vol. 21, no. 2, pp. 193–207, 1998.
- [67] P. Williams, "Jacobi pseudospectral method for solving optimal control problems," *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 2, pp. 293–297, 2004.
- [68] L. Breger and J. How, "Formation flying control for the mms mission using gve-based mpc," in *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*, pp. 565–570, 2005.
- [69] F. Fahroo and I. M. Ross, "On discrete-time optimality conditions for pseudospectral methods," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, pp. 21–24, 2006.

- [70] N. E. Smith, R. G. Cobb, S. J. Pierce, and V. M. Raska, "Optimal Collision Avoidance Trajectories for Unmanned/Remotely Piloted Aircraft," in *AIAA Guidance, Navigation, and Control Conference. Boston, Massachusetts, 19 - 22 August 2013*, AIAA, August 2013.
- [71] D. Benson, *A Gauss Pseudospectral Transcription for Optimal Control*. PhD thesis, Massachusetts Institute of Technology, 2005.
- [72] T. R. Jorris, "Common Aero Vehicle Autonomous Reentry Trajectory Optimization Satisfying Waypoint and No-Fly Zone Constraints," Ph.D. Dissertation ADA472301, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 2007.
- [73] D. Rathbun, S. Kragelund, A. Pongpunwattana, and B. Capozzi, "An Evolution Based Path Planning Algorithm for Autonomous Motion of a UAV Through Uncertain Environments," in *Digital Avionics Systems Conference, 2002. Proceedings. The 21st*, vol. 2, pp. 8D2-1-8D2-12 vol.2, 2002.
- [74] E. Frew, J. Langelaan, and S. Joo, "Adaptive Receding Horizon Control for Vision-Based Navigation of Small Unmanned Aircraft," in *American Control Conference*, 2006.
- [75] R. Padhi, "Optimal Control Formulation using Calculus of Variations." at <http://nptel.iitm.ac.in/courses/101108047>, Department of Aerospace Engineering Indian Institute of Science - Bangalore, August 2012.
- [76] A. V. Rao, D. A. Benson, C. Darby, M. A. Patterson, C. Francolin, I. Sanders, and G. T. Huntington, "Algorithm 902: GPOPS, A Matlab Software for Solving Multiple-Phase Optimal Control Problems Using the Gauss Pseudospectral Method," *ACM Transactions on Mathematical Software (TOMS)*, vol. 37, no. 2, p. 22, 2010.
- [77] J. Shen, T. Tang, and L.-L. Wang, *Spectral Methods: Algorithms, Analysis and Applications*, vol. 41. Springer, 2011.
- [78] M. A. Patterson and A. V. Rao, *GPOPS - II Version 1.0: A General-Purpose MATLAB Toolbox for Solving Optimal Control Problems Using the Radau Pseudospectral Method*. University of Florida, Gainesville, FL 32611-6250, January 2013.
- [79] M. J. Todd and E. A. Yildirim, "On Khachiyan's Algorithm for the Computation of Minimum-Volume Enclosing Ellipsoids," *Discrete Applied Mathematics*, vol. 155, no. 13, pp. 1731-1744, 2007.
- [80] N. E. Smith, R. G. Cobb, S. J. Pierce, and V. M. Raska, "Optimal Collision Avoidance Trajectories via Direct Orthogonal Collocation for Unmanned/Remotely Piloted Aircraft Sense and Avoid Operations," in *AIAA SciTech 2014 Conference. National Harbor, Maryland*, January 2014.
- [81] G. A. Watson and W. D. Blair, "IMM Algorithm for Tracking Targets That Maneuver Through Coordinated Turns," in *Aerospace Sensing*, pp. 236-247, International Society for Optics and Photonics, 1992.

- [82] J. Hartikainen, A. Solin, and S. Sarkka, “Optimal Filtering with Kalman Filters and Smoothers: A Manual for the Matlab toolbox EKF/UKF, Version 1.3,” *Department of Biomedical Engineering and Computational Science, Aalto University School of Science Espoo, Finland*, August 2011.
- [83] X. Li and V. Jilkov, “Survey of Maneuvering Target Tracking. Part I. Dynamic Models,” *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 39, pp. 1333–1364, Oct 2003.
- [84] M. Kochenderfer, J. Kuchar, L. Espindle, and J. Griffith, “Uncorrelated Encounter Model of the National Airspace System, Version 1.0,” tech. rep., DTIC Document, 2008.
- [85] K. Shoemake, “Animating Rotation with Quaternion Curves,” *SIGGRAPH Comput. Graph.*, vol. 19, pp. 245–254, July 1985.
- [86] N. Moshtagh, “Minimum Volume Enclosing Ellipsoid,” *Convex Optimization*, 2005.
- [87] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, “The Quickhull Algorithm for Convex Hulls,” *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, pp. 469–483, 1996.
- [88] J. E. Smith, “Attitude Model of a Reaction Wheel/Fixed Thruster Based Satellite Using Telemetry Data,” Master’s thesis, Air Force Institute of Technology, 2005.
- [89] B. Wie, *Space Vehicle Dynamics and Control Second Edition*. Education Series, AIAA, 2008.
- [90] C. K. Lai and J. Whidborne, “Real-time Trajectory Generation for Collision Avoidance with Obstacle Uncertainty,” in *AIAA Guidance, Navigation, and Control Conference. 8 - 11 August 2011, Portland, Oregon*, AIAA, 2011.
- [91] Smith Nathan E., Arendt Christopher D., Cobb Richard G., “Implementing Conditional Inequality Constraints for Optimal Collision Avoidance,” 2014. Unpublished manuscript intended for archival journal submission.
- [92] C. D. Arendt, “Optimal Control of Fully Routed Air Traffic in the Presence of Uncertainty and Kinodynamic Constraints,” Ph.D. Dissertation, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 2014.
- [93] F. Borrelli, D. Subramanian, A. U. Raghunathan, and L. T. Biegler, “Milp and nlp techniques for centralized trajectory planning of multiple unmanned air vehicles,” in *American Control Conference, 2006*, pp. 6–pp, IEEE, 2006.
- [94] W. Winston and J. Goldberg, *Operations Research: Applications and Algorithms*. Thomson Brooks/Cole, 2004.
- [95] M. Hintermüller, K. Ito, and K. Kunisch, “The Primal-Dual Active Set Strategy as a Semismooth Newton Method,” *SIAM Journal on Optimization*, vol. 13, no. 3, pp. 865–888, 2002.

- [96] A. Sadosky, D. Davis, and D. Isaacson, “Optimal Routing and Control of Multiple Agents Moving in a Transportation Network and Subject to an Arrival Schedule and Separation Constraints,” tech. rep., Technical Memorandum NASA/TM-2012-216032, NASA, Ames Research Center, Moffett Field, CA 94035-0001, USA, 2012.
- [97] J. Ren, K. A. McIsaac, R. V. Patel, and T. M. Peters, “A Potential Field Model Using Generalized Sigmoid Functions,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 2, pp. 477–484, 2007.
- [98] E. W. Weisstein, *CRC Concise Encyclopedia of Mathematics, Second Edition*, pp. 2899–2900. Chapman and Hall, 2 ed., 2002.
- [99] O. J. Farrell and B. Ross, *Solved Problems in Analysis: As Applied to Gamma, Beta, Legendre and Bessel Functions (Dover Books on Mathematics)*, pp. 30–34. Dover Publications, reprint ed., 11 2013.
- [100] D. G. Luenberger, *Optimization by Vector Space Methods*, pp. 29–37. John Wiley & Sons, 1968.
- [101] M. A. Patterson and A. V. Rao, “Gpops- ii: A matlab software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming,” *ACM Transactions on Mathematical Software*, vol. 39, no. 3, 2013.
- [102] J. H. Holland, “Genetic Algorithms,” *Scientific American*, vol. 267, no. 1, pp. 66–72, 1992.
- [103] J. Kennedy, “Particle Swarm Optimization,” in *Encyclopedia of Machine Learning*, pp. 760–766, Springer, 2010.
- [104] N. Sweeney, “Air-to-Air Missile Vector Scoring,” Thesis AFIT/GE/ENG/12-38, Air Force Institute of Technology, March 2012.
- [105] P. S. Maybeck, *Stochastic Models, Estimation, and Control*, vol. 1. Navtech Book and Software Store, 1994.
- [106] C. F. Van Loan, “Computing Integrals Involving the Matrix Exponential,” *IEEE Transaction on Automatic Control*, vol. AC-23, pp. 395 – 404, June 1978.
- [107] R. G. Brown and P. Y. Hwang, *Introduction to Random Signals and Applied Kalman Filtering*. John Wiley and Sons, 3rd ed., 1997.
- [108] S. J. Julier and J. K. Uhlmann, “New Extension of the Kalman Filter to Nonlinear Systems,” in *AeroSense’97*, pp. 182–193, International Society for Optics and Photonics, 1997.
- [109] E. Wan and R. Van der Merwe, “The Unscented Kalman Filter for Nonlinear Estimation,” in *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pp. 153–158, 2000.
- [110] F. Gustafsson, “Particle Filter Theory and Practice with Positioning Applications,” *Aerospace and Electronic Systems Magazine, IEEE*, vol. 25, no. 7, pp. 53–82, 2010.

- [111] N. J. Higham, “Analysis of the Cholesky Decomposition of a Semi-Definite Matrix,” 1990.
- [112] J. B. Burl, *Linear Optimal Control: $H(2)$ and $H(\infty)$ Methods*. Addison-Wesley Longman Publishing Co., Inc., 1998.
- [113] M. J. Eilders, “Decentralized Riemannian Particle Filtering with Applications to Multi-Agent Localization,” Ph.D. Dissertation ADA562462, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, June 2012.
- [114] J. W. Perram and M. Wertheim, “Statistical Mechanics of Hard Ellipsoids. I. Overlap Algorithm and the Contact Function,” *Journal of Computational Physics*, vol. 58, no. 3, pp. 409 – 416, 1985.
- [115] X. Zheng and P. Palffy-Muhoray, “Distance of Closest Approach of Two Arbitrary Hard Ellipses in Two Dimensions,” *Physical Review E*, vol. 75, no. 6, p. 061709, 2007.
- [116] A. H. Barr, “Superquadrics and Angle-Preserving Transformations,” *IEEE Computer Graphics and Applications*, vol. 1, no. 1, pp. 11–23, 1981.
- [117] L. Pallottino, E. M. Feron, and A. Bicchi, “Conflict resolution problems for air traffic management systems solved with mixed integer programming,” *Intelligent Transportation Systems, IEEE Transactions on*, vol. 3, no. 1, pp. 3–11, 2002.
- [118] A. Richards and J. How, “Aircraft trajectory planning with collision avoidance using mixed integer linear programming,” in *American Control Conference, 2002. Proceedings of the 2002*, vol. 3, pp. 1936–1941 vol.3, 2002.
- [119] J. Bellingham, Y. Kuwata, and J. How, “Stable receding horizon trajectory control for complex environments,” in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2003.
- [120] M. Christodoulou and C. Costoulakis, “Nonlinear Mixed Integer Programming for Aircraft Collision Avoidance in Free Flight,” in *Electrotechnical Conference, 2004. MELECON 2004. Proceedings of the 12th IEEE Mediterranean*, vol. 1, pp. 327–330 Vol.1, 2004.
- [121] A. Alonso-Ayuso, L. F. Escudero, and F. Martín-Campo, “Collision Avoidance in Air Traffic Management: A Mixed-Integer Linear Optimization Approach,” *Intelligent Transportation Systems, IEEE Transactions on*, vol. 12, no. 1, pp. 47–57, 2011.
- [122] E. Frazzoli, Z. H. Mao, J. H. Oh, and E. Feron, “Resolution of conflicts involving many aircraft via semidefinite programming,” *Journal of Guidance, Control, and Dynamics*, vol. 24, pp. 79–86, 2013/04/18 2001.
- [123] M. J. Kochenderfer and J. Chryssanthacopoulos, “Robust airborne collision avoidance through dynamic programming,” *Massachusetts Institute of Technology Lincoln Laboratory. Project Report ATC-371*, 2011.
- [124] N. Durand, J.-M. Alliot, and F. Médioni, “Neural nets trained by genetic algorithms for collision avoidance,” *Applied Intelligence*, vol. 13, no. 3, pp. 205–213, 2000.

- [125] L. Singh and J. Fuller, "Trajectory generation for a uav in urban terrain, using nonlinear mpc," in *American Control Conference, 2001. Proceedings of the 2001*, vol. 3, pp. 2301–2308, IEEE, 2001.
- [126] D. Shim, H. Kim, and S. Sastry, "Decentralized nonlinear model predictive control of multiple flying robots," in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol. 4, pp. 3621–3626 vol.4, 2003.
- [127] A. Richards and J. How, "Decentralized model predictive control of cooperating uavs," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 4, pp. 4286–4291 Vol.4, 2004.
- [128] D. H. Shim, H. Chung, and S. S. Sastry, "Conflict-free navigation in unknown urban environments," *Robotics & Automation Magazine, IEEE*, vol. 13, no. 3, pp. 27–33, 2006.
- [129] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment: A collision cone approach," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 28, no. 5, pp. 562–574, 1998.
- [130] J. Goss, R. Rajvanshi, and K. Subbarao, *Aircraft Conflict Detection and Resolution Using Mixed Geometric And Collision Cone Approaches*. American Institute of Aeronautics and Astronautics, 2013/04/08 2004.
- [131] A. L. Smith, "UAS Collision Avoidance Algorithm that Minimizes the Impact on Route Surveillance," tech. rep., DTIC Document, 2009.
- [132] A. Mujumdar and R. Padhi, *Nonlinear Geometric Guidance and Differential Geometric Guidance of UAVs for Reactive Collision Avoidance*. American Institute of Aeronautics and Astronautics, 2013/04/08 2010.
- [133] K. Bilimoria, *A geometric optimization approach to aircraft conflict resolution*. American Institute of Aeronautics and Astronautics, 2013/04/08 2000.
- [134] C. Carbone, U. Ciniglio, F. Corrado, and S. Luongo, "A novel 3d geometric algorithm for aircraft autonomous collision avoidance," in *Decision and Control, 2006 45th IEEE Conference on*, pp. 1580–1585, IEEE, 2006.
- [135] G. Fasano, D. Accardo, A. Moccia, C. Carbone, U. Ciniglio, F. Corrado, and S. Luongo, "Multi-sensor-based fully autonomous non-cooperative collision avoidance system for unmanned air vehicles," *Journal of Aerospace Computing, Information, and Communication*, vol. 5, pp. 338–360, 2013/04/08 2008.
- [136] J.-W. Park, H.-D. Oh, and M.-J. Tahk, "UAV Collision Avoidance Based on Geometric Approach," in *SICE Annual Conference, 2008*, pp. 2122–2126, 2008.
- [137] S. Temizer, M. Kochenderfer, L. Kaelbling, T. Lozano-Perez, and J. Kuchar, *Collision Avoidance for Unmanned Aircraft using Markov Decision Processes*. American Institute of Aeronautics and Astronautics, 2010.

- [138] I. Hwang and C. Tomlin, "Protocol-based conflict resolution for finite information horizon," in *American Control Conference, 2002. Proceedings of the 2002*, vol. 1, pp. 748–753, IEEE, 2002.
- [139] J. K. Archibald, J. C. Hill, N. A. Jepsen, W. C. Stirling, and R. L. Frost, "A satisficing approach to aircraft conflict resolution," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 38, no. 4, pp. 510–521, 2008.
- [140] E. Besada-Portas, L. de la Torre, J. De La Cruz, and B. de Andrés-Toro, "Evolutionary Trajectory Planner for Multiple UAVs in Realistic Scenarios," *Robotics, IEEE Transactions on*, vol. 26, no. 4, pp. 619–634, 2010.
- [141] N. Durand, J.-M. Alliot, and J. Noailles, "Automatic aircraft conflict resolution using genetic algorithms," in *Proceedings of the 1996 ACM symposium on Applied Computing*, pp. 289–298, ACM, 1996.
- [142] N. Kantas, A. Lecchini-Visintini, and J. Maciejowski, "Simulation-based Bayesian Optimal Design of Aircraft Trajectories for Air Traffic Management," *International Journal of Adaptive Control and Signal Processing*, vol. 24, no. 10, pp. 882–899, 2010.
- [143] R. Karlsson, J. Jansson, and F. Gustafsson, "Model-based statistical tracking and decision making for collision avoidance application," in *American Control Conference, 2004. Proceedings of the 2004*, vol. 4, pp. 3435–3440 vol.4, 2004.
- [144] A. Lecchini, W. Glover, J. Lygeros, and J. Maciejowski, *Monte Carlo Optimization Strategies for Air-Traffic Control*. American Institute of Aeronautics and Astronautics, 2013/04/18 2005.
- [145] M. Prandini, J. Hu, J. Lygeros, and S. Sastry, "A probabilistic approach to aircraft conflict detection," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 1, no. 4, pp. 199–220, 2000.
- [146] L. Peng and Y. Lin, "Study on the model for horizontal escape maneuvers in tcas," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 11, no. 2, pp. 392–398, 2010.
- [147] T. Dong, X. Liao, R. Zhang, Z. Sun, and Y. Song, "Path Tracking and Obstacles Avoidance of UAVs-Fuzzy Logic Approach," in *Fuzzy Systems, 2005. FUZZ'05. The 14th IEEE International Conference on*, pp. 43–48, IEEE, 2005.
- [148] S.-C. Han and H. Bang, "Proportional navigation-based optimal collision avoidance for UAVs," in *2nd International Conference on Autonomous Robots and Agents*, pp. 13–15, 2004.
- [149] C. Tomlin, G. Pappas, and S. Sastry, "Conflict resolution for air traffic management: a study in multiagent hybrid systems," *Automatic Control, IEEE Transactions on*, vol. 43, no. 4, pp. 509–521, 1998.
- [150] E. Crück and J. Lygeros, "Sense and Avoid System for a MALE UAV," in *proceedings of the AIAA Conference on Guidance, Navigation and Control*, 2007.

- [151] P. K. Menon, G. D. Sweriduk, and B. Sridhar, "Optimal strategies for free-flight air traffic conflict resolution," *Journal of Guidance, Control, and Dynamics*, vol. 22, pp. 202–211, 2013/04/05 1999.
- [152] S. Shandy and J. Valasek, *Intelligent agent for aircraft collision avoidance*. American Institute of Aeronautics and Astronautics, August 2001.
- [153] A. Bicchi and L. Pallottino, "On optimal cooperative conflict resolution for air traffic management systems," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 1, no. 4, pp. 221–231, 2000.
- [154] B. R. Geiger, J. F. Horn, A. M. DeLullo, L. N. Long, and A. F. Niessner, "Optimal Path Planning of UAVs Using Direct Collocation with Nonlinear Programming," *AIAA Paper*, no. 2006-6199, 2006.
- [155] L. Lewis, I. Ross, and Q. Gong, "Pseudospectral Motion Planning Techniques for Autonomous Obstacle Avoidance," in *Decision and Control, 2007 46th IEEE Conference on*, pp. 5997–6002, 2007.
- [156] J. Zhang and S. Sastry, "Aircraft Conflict Resolution: Lie-Poisson Reduction for Game on SE (2)," in *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*, vol. 2, pp. 1663–1668, IEEE, 2001.
- [157] J. Hu, M. Prandini, and S. Sastry, "Optimal Maneuver for Multiple Aircraft Conflict Resolution: A Braid Point of View," in *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, vol. 4, pp. 4164–4169, IEEE, 2000.
- [158] H. H. Versteegt and H. G. Visser, *Traffic Complexity Based Conflict Resolution*. American Institute of Aeronautics and Astronautics, 2013/04/05 2002.
- [159] R. Ghosh and C. Tomlin, "Maneuver Design for Multiple Aircraft Conflict Resolution," in *American Control Conference, 2000. Proceedings of the 2000*, vol. 1, pp. 672–676 vol.1, 2000.
- [160] E. V. Denardo, *Dynamic programming: models and applications*. Courier Dover Publications, 1982.
- [161] K. Y. Lee and M. A. El-Sharkawi, *Modern Heuristic Optimization Techniques: Theory and Applications to Power Systems*, vol. 39. Wiley-IEEE press, 2008.
- [162] J. Krozel and M. Peters, "Strategic conflict detection and resolution for free flight," in *Decision and Control, 1997., Proceedings of the 36th IEEE Conference on*, vol. 2, pp. 1822–1828 vol.2, 1997.
- [163] J. Sumita, *Autonomous NAS Flight Control for UAV by Fuzzy Concept*. American Institute of Aeronautics and Astronautics, 2004.
- [164] D. Sislak, P. Volf, and M. Pechoucek, "Agent-Based Cooperative Decentralized Airplane-Collision Avoidance," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 12, no. 1, pp. 36–46, 2011.

REPORT DOCUMENTATION PAGE			<i>Form Approved</i> OMB No. 0704-0188	
<p>The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</p>				
1. REPORT DATE (DD-MM-YYYY) 26-12-2014		2. REPORT TYPE Doctoral Dissertation		3. DATES COVERED (From — To) Jan 2012–Oct 2014
4. TITLE AND SUBTITLE Optimal Collision Avoidance Trajectories for Unmanned/Remotely Piloted Aircraft		5a. CONTRACT NUMBER		
		5b. GRANT NUMBER		
		5c. PROGRAM ELEMENT NUMBER		
6. AUTHOR(S) Smith, Nathan E., Colonel, USAF		5d. PROJECT NUMBER		
		5e. TASK NUMBER		
		5f. WORK UNIT NUMBER		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of 2950 Hobson Way WPAFB OH 45433-7765		8. PERFORMING ORGANIZATION REPORT NUMBER AFIT-ENY-DS-14-D-34		
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory Aerospace Systems Directorate (AFRL/RQ) Attn: Vincent Raska 2210 Eighth Street, B20146, R122 WPAFB, OH 45433-7765 vincent.raska@us.af.mil , (937) 938-4642		10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/RQQC		
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)		
12. DISTRIBUTION / AVAILABILITY STATEMENT DISTRIBUTION STATEMENT A: APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED				
13. SUPPLEMENTARY NOTES This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.				
14. ABSTRACT The post-911 environment has punctuated the force-multiplying capabilities that Remotely Piloted Aircraft (RPA) provides combatant commanders at all echelons on the battlefield. Not only have unmanned aircraft systems made near-revolutionary impacts on the battlefield, their utility and proliferation in law enforcement, homeland security, humanitarian operations, and commercial applications have likewise increased at a rapid rate. As such, under the Federal Aviation Administration (FAA) Modernization and Reform Act of 2012, the United States Congress tasked the FAA to provide for the safe integration of civil unmanned aircraft systems into the national airspace system (NAS) as soon as practicable, but not later than September 30, 2015. However, a necessary entrance criterion to operate RPAs in the NAS is the ability to Sense and Avoid (SAA) both cooperative and non-cooperative air traffic to attain a target level of safety as a traditional manned aircraft platform. The goal of this research effort is twofold: First, develop techniques for calculating optimal avoidance trajectories, and second, develop techniques for estimating an intruder aircraft's trajectory in a stochastic environment. This dissertation describes the optimal control problem associated with SAA and uses a direct orthogonal collocation method to solve this problem and then analyzes these results for different collision avoidance scenarios.				
15. SUBJECT TERMS Collision avoidance, unmanned or remotely piloted aircraft, optimal control, stochastic estimation, direct orthogonal collocation				
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 263
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U		
			19a. NAME OF RESPONSIBLE PERSON Dr. Richard G. Cobb (ENY)	
			19b. TELEPHONE NUMBER (Include Area Code) (937) 255-3636 x4559 Richard.Cobb@afit.edu	